DIGITAL VIDEO STABILIZATION

LIU SHUAICHENG

(M.S. SOC, NUS, 2010)

(B.Sc. SICHUAN UNIVERSITY, 2008)

A THESIS SUBMITTED

FOR THE DEGREE OF DOCTOR OF PHILOSOPHY

DEPARTMENT OF ELECTRICAL AND COMPUTER ENGINEERING

NATIONAL UNIVERSITY OF SINGAPORE

2014

To my parents...

Declaration

I hereby declare that this thesis is my original work and it has been written by me in its entirety. I have duly acknowledged all the sources of information which have been used in the thesis.

This thesis has also not been submitted for any degree in any university previously.

Liu Shuaicheng Nov. 28 2014

Shuaicheng Liu

November 28, 2014

Acknowledgements

I am deeply grateful to my PhD supervisor Professor Tan Ping for his patient guidance, continued encouragement and support throughout my PhD studies. His wisdom and kindness will always inspire me. I am heartily thanksful to my mentor Dr. Yuan Lu in Microsoft Research Asia, who is always enlightening and supportive throughout my Ph.D trainings. I would like to express my deep grateful to my mentor Dr. Wang Jue in Adobe Research, for his brilliant insights and suggestions on the research projects we have cooperated on. I would also like to thank Professor Michael S.Brown for his research training during my master studies in NUS. I would like to thank my committee members for their valuable critism and suggestion to improve my research work, including this thesis.

I would like to thank an inspiring group of colleagues in the Vision and Machine Learning lab: Jiang Nianjuan, Zou Danping, Zhou Zhenglong, Zhang Yinda, Wu zhe, Huang Rui, Cui Zhaopeng, Lin Kaimo, Wang Peilin, Wang Yinting, Shi Boxin, Jiang Hanqing, Shaojie Zhuo, Liu Yugang, Deng Xiaoming, Shu Bo, Wang Peng and colleagues in Vision and Interactive Media Lab: Li Zhuwen, Guo Jiaming, Zhou Qiang, Luo Ye and too many others to list individually. I thank them for having provided an enjoyable and stimulating lab environment. I really enjoy the collaborations and insightful discussions with them. Besides, I would like to

acknowledge a group of colleagues in Computer Vision lab, School of Computing(SOC): Lu Zheng, Tai Yu-Wing, Gao Junhong, Deng Fanbo, Lin Haiting, Cheng Yuan, Guo Dong, Ye Ning, Shaojie Zhuo, Li Hao, Song Zhiyuan, Wang Yumei etc. Back the days when I pursuing a master degree in SOC NUS, we shared experience which became valuable memories throughout my life.

Finally, I would like to express my deepest gratitude to my parents for their understanding and encouragement throughout the years.

Contents

Summary

This thesis addresses the problem of digital video stabilization. Videos captured by hand-held devices (e.g., cell-phones or DVs) often appear remarkably shaky. Digital video stabilization improves the video quality by removing unwanted camera motions. In this thesis, three different methods are presented. We first address video stabilization by adopting a depth camera. We show that the depth can facilitate both camera motion estimation and frame warping, thus make the video stabilization a much well posed problem. Then, we present a video stabilization approach named as bundled camera paths, in which multiple 2D camera paths are proposed to represent camera motions. Its mesh-based , spatially-variant motion representation allows us to fundamentally handle parallax issues without the help of long feature trajectories. Finally, we present a novel motion model, SteadyFlow, which has per-pixel level accuracy. The SteadyFlow is a specific optical flow by enforcing strong spatial coherence. Our experiments demonstrate the effectiveness of our stabilization methods on real-word challenging videos.

This thesis is organized to begin with an overview on challenges of video stabilization, followed by self-contained chapters for three different methods in video stabilization. A summary chapter is included to summarize our contributions and discuss future work.

List of Figures

Chapter 1

Introduction

Videos captured by hand-held devices(e.g., cell-phones, portable camcorders) often appear remarkably shaky and undirected. Traditionally, a stabilized video is captured by expensive professional devices, a camera mounted on a body of a moving person or placed on a track. Figure 1.1 left shows some examples. Digital video stabilization improves the video quality by processing videos captured by consumer level devices illustrated in Figure 1.1 right.

In general, video stabilization consists of the following three main steps: (1) estimating the camera motion to obtain original shaky camera path, (2) creating a new smooth camera path, and (3) synthesizing the stabilized video using the smoothed camera path. Figure 1.2 shows the pipeline. For the first step, the motion estimation can be either in 3D or 2D. According to the adopted motion model, video stabilization can be categorized as 3D-based[56], 2D-based[64, 39] or 2.5D-based[57, 35] methods. 3D-based methods reconstruct the scene and recover the 3D camera poses using structure from motion(SFM) algorithms[41]. 2D-based methods estimate affine or homography between neighboring frames. Camera path is represented by the concatenation of these

Figure 1.1: Left: Professional videos are often captured by expensive equipments. Right: Digital video stabilization improves video quality captured by hand-held devices.(cell-phones, DVs, tablets)

Figure 1.2: A general pipeline for video stabilization.

linear transformations. 2.5D-based approaches relax the requirement of full 3D reconstruction to some partial 3D information such as the epipolar geometry[35]. A detailed literature review is presented in Chapter 2.

Generally speaking, 2D methods are more robust and faster because they only estimate a linear transformation model between neighboring frames. But the 2D linear motion model is too weak to fundamentally handle the parallax cased by non-trivial depth changes. On the contrary, the 3D methods can deal with parallax in principle and generate strongly stabilized results when 3D reconstruction is feasible. However, the 3D reconstruction is less robust to various degenerations such as feature tracking failure, motion blur, camera zooming, and rapid camera motion. In the following ,we will discuss the challenging issues in video stabilization and demonstrate some com-

mon artifacts when stabilization fails.

1.1 Challenges in Video Stabilization

Video stabilization algorithms have been improved dramatically in recent years. Some works have successfully transferred into commercial softwares, e.g., the Warp stabilizer in Adobe After Effects built on the technique of subspace method[57] and YouTube stabilizer developed from homography mixture model[37]. They can produce very good results on many casual online videos and even handle some difficult examples. However, there are always some challenge cases that go beyond the power of existing techniques. In the following, we first review some of the major challenges in casually shot consumer videos. We further demonstrate the kind of failure they will cause in a video stabilization algorithm. This discussion motivates our design of new stabilization algorithms.

Large depth variation A scene contains depth variation is very common in consumer videos. Figure 1.3 left shows two examples. The reason depth becomes a challenge is mainly due to the plane based motion model. In 2D video stabilization[65], affines or homographies are used to model the motion between neighboring frames. A single homography is valid only when filming a planer scene or the camera under goes pure rotation. The best fitting homography cannot describe all the motions of the scene, resulting the wobble artifacts. Examples of wobble distortion are shown in Figure 1.3 right. Please notice the shearing of the house and the curving of the tree trunk. Recently, there are methods [38, 61] adopting multiple homographies for motion estimation. Even though, large depth variation still remains a challenge issue and

Figure 1.3: Left: two scenes contain large depth variation. Right: stabilized frames suffer from wobble distortions.

requires more research efforts.

Quick camera motion Quick camera motion is another type of challenge for video stabilization,(e.g., quick rotation and zooming). 3D and 2.5D methods rely on long feature trajectories to stabilize a video. However, when quick motion happens, the length of trajectories drop quickly and even approaches zero in some extreme cases. This severely damages the performance of trajectory-based stabilization methods. Figure 1.4 shows two examples with quick camera zooming(top) and quick camera rotation(bottom). The stabilized results contain large empty regions.

Large moving foreground Large dynamic objects can easily confuse a stabilizer during camera motion estimation. If the dynamic object size is small, RANSAC can be adopted for 2D methods and 3D methods, to exclude moving objects. However, it

Figure 1.4: Two examples contain quick camera motion. First row: camera with quick zooming. Second row: camera contains quick rotation. The results contain large empty regions.

is difficult to distinguish the foreground and the background in the presence of large size moving objects. If the foreground motion(fully or partially) is considered as the background camera motion, the error motion would led to jitter and unstable results. Recently, a user-assisted method[4] is proposed to address this issue by letting the user to exclude features on the foreground during motion estimation. Nevertheless, motion segmentation is still a challenge for automatic systems.

Motion blur Camera shake can blur video frames significantly at times when shake is intense. Figure 1.6 shows such two examples where the blurred frames severely damage the quality of the videos. Many stabilization methods can successfully stabilize the video content, however they leave the blurriness cased by original camera motion untouched. On the other hand, stabilization systems rely on feature tracking to

Figure 1.5: The camera motion estimation is inaccurate in the presence of large moving foreground.

Figure 1.6: Video frames are blurred caused by the camera shake. Two examples of blurry frames borrowed from Cho et al.[19]

estimate the camera motion. However, feature tracking over blurry frames is not reliable due to the lack of sharp image features. Thus, handling motion blur[19] becomes an important task for a robust video stabilization system.

Rolling shutter effects The rolling shutter effects[67] are caused by parallel readout scheme of CMOS cameras. Pixels within a row are read out simultaneously, but integration time is shifted row by row, resulting the bending of straight lines in the captured frame. Figure 1.7 shows two examples sufferring from rolling shutter distortion. Many cell-phone cameras employ CMOS sensors due to their low power consumption. It is of great practical importance to handel rolling shutter effects to obtain satisfactory results for video stabilization. This requires non-linear motion models and more so-

Figure 1.7: Rolling shutter effects

phisticated smoothing strategies.

1.2 Objective

Each of the individual challenge is a research problem. A good video stabilization system should try to overcome as much as possible. However, the problem becomes much more difficult when multiple challenges happen together, e.g. rolling shutter effects together with large moving objects or quick camera motion together with motion blur. There is a high possibility that these challenges are linked together in real world scenarios. Empirically, we find that when 3D reconstruction is feasible, 3D methods often produce the best results for scene with large depth variation. However, they lack the ability to handle the other challenges. The 2D based methods, on the other hand, are robust to quick camera motions, but with limited ability for depth handling. A video contains large moving objects require motion segmentation methods[74, 23] to discover the camera motion. However, motion segmentation for shaky videos with dominate foregrounds is tough and challenge.

Another important issue is the stability. Some methods can successfully remove the

high-frequency camera jitters but leave the low-frequency camera shake untouched,(e.g., low-frequency bounces originated from a person walking during the capture). To obtain high quality stabilized videos, we need to also suppress the low-frequency shake with some advanced smoothing approach. Artifacts like wobble and excessive cropping would be introduced if the camera path is not appropriately smoothed. In practice, reducing stability can be considered as a kind of wobble suppression when there is no other way around. Because the original shaky input contains no wobbles at all. One can always suppress the wobbles at the sacrifice of stability. We need to seek a good balance to achieve good stability with a reasonable cropping size and limited wobbles.

In summary, the objective for a desirable video stabilization system contains the following goals: no geometrical distortions and wobbles, good stability, reasonable cropping size, correcting rolling shutter effects , handling motion blur. Although, none of the existing methods can satisfy all the goals, it is worth to explore towards this direction.

1.3 Contributions

This section gives a brief introduction to the problems we have studied: Stabilizing videos with depth cameras and stabilizing videos captured by traditional video recorder (cell-phones, tablets and DVs). The central idea is on how to model camera motion and define smoothing methods properly.

RGBD videos Previous video stabilization methods often employ homographies to model transitions between consecutive frames, or require robust long feature tracks for structure from motion. However, the homography model is invalid for scenes with

significant depth variations, and feature point tracking is fragile in videos with textureless objects, severe occlusion or camera rotation. To address these challenging cases, we propose to solve video stabilization with an additional depth sensor. Though the depth image is noisy, incomplete and low resolution, it facilitates both camera motion estimation and frame warping, which make the video stabilization a much well posed problem. This work has been published in CVPR 2012 [60][1].

Bundled camera paths This method is proposed to address consumer level videos captured by traditional devices(mobile phones,tablets, camcorders). We model the camera motion with a bundle of(multiple) camera paths. The proposed model is derived from a mesh-based, spatially-variant motion representation and an adaptive, space-time path optimization. Our motion representation allows us to fundamentally handle parallax and rolling shutter effects while it does not require long feature trajectories or sparse 3D reconstruction. We introduce the as-similar-as-possible idea to make motion estimation more robust. Our space-time path smoothing adaptively adjusts smoothness strength by considering discontinuities, cropping size and geometrical distortion in a unified optimization framework. The evaluation on a large variety of consumer videos demonstrates the merits of our method. This work has been published in SIGGRAPH 2013[61][2].

SteadyFlow This method is also targeted on videos captured by traditional devices. We propose a novel motion model, SteadyFlow, to represent the motion between neighboring video frames for stabilization. A SteadyFlow is a specific optical flow by enforcing strong spatial coherence, such that smoothing feature trajectories can be re-

[1]Project page: http://www.liushuaicheng.org/CVPR2012/index.htm
[2]Project page: http://www.liushuaicheng.org/SIGGRAPH2013/index.htm

placed by smoothing pixel profiles, which are motion vectors collected at the same pixel location in the SteadyFlow over time. In this way, we can avoid brittle feature tracking in a video stabilization system. Besides, SteadyFlow is a more general 2D motion model which can deal with spatially-variant motion. We initialize the SteadyFlow by optical flow and then discard discontinuous motions by a spatial-temporal analysis and fill in missing regions by motion completion. Our experiments demonstrate the effectiveness of our stabilization on real-world challenging videos. This work has been published in CVPR 2014[62][1].

1.4 Thesis Organization

The remainder of this thesis is organized as follows. Chapter 2 reviews the previous works on video stabilization. Chapter 3 presents the work of video stabilization with a depth camera. Chapter 4 discusses bundled camera paths. Chapter 5 details our work on SteadyFlow for video stabilization. Chapter 6 summarizes the thesis. Chapter 3,4,5 are self-contained. Each chapter describes a video stabilization method. Chapter 6 concludes this thesis with a discussion of limitations and some future research directions.

[1]Project page: http://www.liushuaicheng.org/CVPR2014/index.htm

Chapter 2

Literature Review

According to the adopted motion model, video stabilization can be categorized into 3D, 2D and 2.5D methods. Besides, the handling of rolling shutter effects is highly related to the video stabilization.

3D methods 3D methods require explicit 3D structures for video stabilization, including 3D camera poses and scene depth. These structures can be obtained by the SFM algorithms[92, 90, 22, 79] or by the adopting of depth sensors[76]. Given the original shaky 3D camera path, a smoothed virtual camera path is recovered. The stabilized video is rendered along the virtual path as if it were taken from this new path. This rendering process is often referred to as novel view synthesis. When 3D reconstruction is feasible, it often produces the highest quality of results due to its physical correctness. Our work "Video stabilization with a depth camera[60]" belongs to this category.

2D methods 2D methods estimate 2D transformations between consecutive video frames. By concatenating these transformations, camera path in 2D space is obtained.

The stabilized video is generated by smoothing the 2D camera path. The 2D transformations are often affines or homographies. The research focus is given to both motion estimation[65] and path planning[32, 33]. Strictly speaking, a homography is only valid when the scene is planer or the camera undergoes purely rotational motion[41]. When a scene contains large depth variations, the 2D model is invalid. Artifacts such as content distortions would be introduced in the stabilized results. On the other hand, the advantage of 2D methods is the robustness. It only requires feature correspondences between neighboring frames. The 2D model fitting is much more robust compared with 3D reconstruction. Our work "Bundled camera paths for video stabilization[61]" and "Steadyflow: spatially smooth optical flow for video stabilization[62]" belong to this category. Our methods can produce results with the same quality as 3D methods while enjoy the robustness of 2D methods.

2.5D methods 2.5D methods relax the requirement of full 3D reconstruction to some partial 3D information(e.g., epipolar geometry [35]). The 3D information is embedded in the feature trajectories. The 2.5D methods argue that the 3D reconstruction is overshoot for video stabilization purpose. The 2.5D methods can produce comparable results as full 3D methods while reduce the computational costs. However, the requirement of long feature trackings(e.g., feature trackings for 30 consecutive frames) is still a bottleneck for the robustness.

Rolling Shutter Rolling shutter removal and video stabilization are highly related. The rolling shutter effects [67] are caused by parallel readout scheme of CMOS sensors. Pixels within a row are read out simultaneously, but integration time is shifted row by row, resulting in bending of straight lines in the captured image. Many cell-phone

cameras employ CMOS sensors due to their low power consumption. Stabilizing a video alone would not produce satisfactory result if the video also suffers from rolling shutter effects. Both our work [61] and [62] can rectify the rolling shutter effects.

In the following sections, we briefly review the prior works based on the categories. We highlight one representative work for each category.

2.1 3D Video Stabilization

3D methods estimate 3D camera motion for stabilization. Beuhler et al. [15] proposed a 3D video stabilization method based on a projective reconstruction of the scene with an uncalibrated camera. When Euclidean reconstruction is feasible, Zhang et al. [76] smoothed the camera trajectories to minimize its acceleration in rotation, translation and zooming. Liu et al. [56] proposed a full 3D stabilization method by introducing content-preserving warps(IPW) for the novel view synthesis. Zhou et al.[96] further extended the content-preserving warps with plan-based constraints. These methods are more or less limited by their adopted 3D reconstruction algorithms. Though there is significant progress [70, 1, 27, 29, 45, 46, 86] in 3D reconstruction, reconstructing a general video is still difficult. In the following, we briefly review the method of content-preserving warp.

Content-Preserving Warp

Liu et al.[56] proposed the content-preserving warp for the novel view synthesis. It is inspired by as-rigid-as-possible shape manipulation [42]. Given the input video frame $\hat{I}_t$, the corresponding output video frame I_t is generated by a warp from $\hat{I}_t$. 3D reconstruction provides a sparse set of 3D points. They can be projected onto both

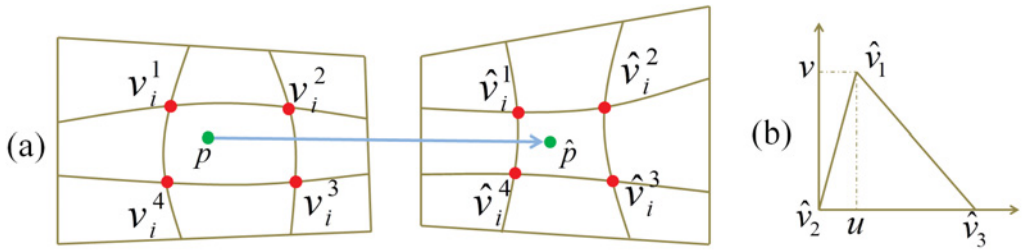


Figure 2.1: (a) A pair of matched features (p,$\hat{p}$) should be represented by the same set of bilinear interpolation weights of their four enclosing vertices. (b) The smooth term requires each triangle $\hat{v_1}$,$\hat{v_2}$,$\hat{v_3}$ to follow a similarity transformation.

the input and output cameras, yielding two sets of corresponding 2D points: $\hat{P}$ on the input frame and P on the output frame.

data term Suppose $\{p, \hat{p}\}$ is the p-th matched feature pair from input and output frame respectively. The feature p can be represented by a 2D bilinear interpolation of the four vertices $V_p = [v_p^1, v_p^2, v_p^3, v_p^4]$ of the enclosing grid cell: $p = V_p w_p$, where $w_p = [w_p^1, w_p^2, w_p^3, w_p^4]^\top$ are interpolation weights that sum to 1. The corresponding feature $\hat{p}$ can be represented by the same weights of the warped grid vertices $\hat{V}_p = [\hat{v}_p^1, \hat{v}_p^2, \hat{v}_p^3, \hat{v}_p^4]$. Figure 3.6 (a) shows the relationship. Therefore the data term is defined as

$$E_d(\hat{V}) = \sum\nolimits_p ||\hat{V}_p w_p - \hat{p}||^2. \tag{2.1}$$

Here $\hat{V}$ contains all the warped grid vertices.

similarity transformation term As illustrated in Figure 3.6 (b), the similarity term is defined as:

$$E_s(\hat{V}) = \sum_{\hat{v}} \|\hat{v} - \hat{v}_1 - sR_{90}(\hat{v}_0 - \hat{v}_1)\|^2,\ R_{90} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \tag{2.2}$$

where $s = \|v - v_1\|/\|v_0 - v_1\|$ is a known scalar computed from the initial mesh. This similarity transformation term requires the triangle of neighboring vertices v, v_0, v_1 undergoes a similarity transformation.

The final energy $E(\hat{V})$ is obtained by combining two terms.

$$E(\hat{V}) = E_d(\hat{V}) + \alpha E_s(\hat{V}), \tag{2.3}$$

where α is a weight to control the amount of regularization. This energy equation is quadratic and can be minimized by solving a sparse linear system. Content preserving warp is applied to warp a frame to its novel view point. It shows greater advantage over traditional image based rendering techniques[15, 20].

2.2 2D Video Stabilization

2D stabilization methods use a series of 2D transformations (such as homography or affine transformations) to represent the camera motion, and smooth these transformations to stabilize the video. Early 2D video stabilization methods such as[66, 64, 49] estimated affine transformations or homographies between consecutive frames and applied low pass filtering to reduce high frequency camera jitters. To suppress low frequency camera shakes, Chen et al.[17] fits polynomial curves to camera trajectories. Gleicher and Liu [32] further broke camera trajectories into segments and fitted smooth motion to each of them for better camera motion. More recently, Grundmann et al. [39] applied cinematography rules[33] and represented camera motion by a combination of constant, linear or parabolic motion. This technique has been integrated into Google YouTube. It is robust, follows cinematography rules, and works well on many casual

online videos.

Let the video be a sequence of images I_1,I_2,...I_n, where each frame pair (I_{t-1}, I_t) is associated with a linear motion model F_t. The camera path C_t is defined as:

$$C_{t+1} = C_t F_{t+1} \Rightarrow C_t = F_1 F_2 ... F_t. \tag{2.4}$$

L1-base Cinematography Camera Path

From a cinematographic viewpoint, a pleasant steady viewing experience is conveyed by the use of static cameras, panning cameras mounted on tripods and cameras placed onto a dolly[37]. To mimic professional footage, the optimized camera paths should be composed by the following path segments:

- A constant path, representing a static camera
- A path of constant velocity, representing a panning or a dolly shot.
- A path of constant acceleration, representing the ease-in and out transition between static and panning cameras.

To obtain the optimal path, Grundmann et al. [39] formulated the problem as a constrained L_1 minimization. Given the original path C_t, the desired optimal path is denoted as:

$$P_t = C_t B_t \tag{2.5}$$

where $B_t = C_t^{-1} P_t$ is the update transform which brings the original frame to its stabilized position. The objective function is formulated as:

$$O(P) = w_1 |D(P)|_1 + w_2 |D^2(P)|_1 + w_3 |D^3(P)|_1 \tag{2.6}$$

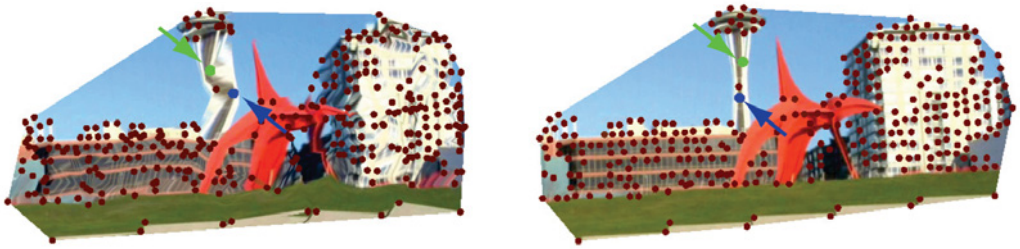
This objective function can be minimized by linear programming.

2.3 2.5D Video Stabilization

A trade-off between 2D and 3D stabilization techniques is to directly smooth the trajectories of tracked image feature points. Goldstein and Fattal [35] utilized an "epipolar transfer" technique to avoid the fragile 3D reconstruction. Wang et al.[85] represented each trajectory as a Bezier curve and smoothed with a spatial-temporal optimization. To address the occlusion issue, Lee et al. [50] introduced feature pruning to choose robust feature trajectories for smoothing. Liu et al. [57] smoothed some basis trajectories of the subspace [43] extracted from the feature tracks (preferably longer than 50 frames). This method achieves similar quality to the full 3D methods, while reducing the requirement from 3D reconstruction to long feature trajectories. It has been transferred to Adobe After Effects as a video stabilization function named "Warp Stabilizer". Recently, Liu et al.[58] extended the subspace method to deal with stereoscopic videos. The core ideas of subspace [57] is one of the representative work in 2.5D methods.

Subspace Video Stabilization

Given a set of 2D point trajectories, we seek to find the appropriate positions for these points at the output frame to stabilize the video. The trajectories can be concatenated



Filter each trajectory independently

Filter the eigen-trajectories

Figure 2.2: Subspace Low-path filtering. Left: filter each trajectory independently introduce artifacts as ignoring of 3D information. Right: filter eigen-trajecotries in the subspace. The figures are borrowed from [56].

into a trajectory matrix M:

$$M_{2N\times F} = \begin{bmatrix} x_1^1 & x_2^1 & \dots & x_F^1 \\ y_1^1 & y_2^1 & \dots & y_F^1 \\ & & . & \\ x_1^N & x_2^N & \dots & x_F^N \\ y_1^N & y_2^N & \dots & y_F^N \end{bmatrix} \tag{2.7}$$

with N features per frame and F frames in total. If a low-pass filter is directly applied to this matrix, distortion would happen as independently smoothing feature trajectories breakdown the relationship between points. Figure 2.2 left shows such an example. To maintain this relationship during the smoothing, a subspace constraint is proposed. In general, motion trajectories from a perspective camera will lie on a non-linear manifold[81, 34]. It is possible to approximate the manifold locally with a linear subspace. Irani [44] showed that the trajectory matrix should have at most rank 9. This low-rank constraint implied that the trajectory matrix M can be factored into

the product of two low-rank matrices:

$$M_{2n\times k} \approx W \odot (C_{2n\times r} E_{r\times k}) \tag{2.8}$$

where W is a binary mask matrix indicating missing data, and $\odot$ means component-wise multiplication. E is the eigen-trajectories and C contains the coefficient for the linear combination. If we apply a smooth operation K, it can be further derived:

$$\hat{M} = W \odot (CE)K = W \odot C(EK) = W \odot C\hat{E} \tag{2.9}$$

which means we first filtering the eigen-trajectories E to obtain $\hat{E}$, and then obtain a new sub matrix $\hat{M}_{2n\times k}$ by multiplying $\hat{E}$ with the original coefficient matrix C. Output frames can be obtained by content-preserving warp guided by the control points in M and $\hat{M}$. Figure 2.2 right shows an example. With the subspace constraint, the relationship between features are appropriately preserved.

2.4 Rolling Shutter

Rolling shutter methods estimate and correct inter-row motion caused by the row-parallel readout. Prior works designed different parametric inter-row motion models, including a per-row translation model [51, 5] and 3D rotation model [25, 26]. Karpenko et al.[49, 40] used dedicated hardware – the gyroscope on mobile devices, to correct the rolling shutter effects in real-time. Recently, Grundmann et al.[38] proposed a calibration-free homography mixture model, which shows significant improvement. In the following, we introduce the work of Grundmann et al.[38].

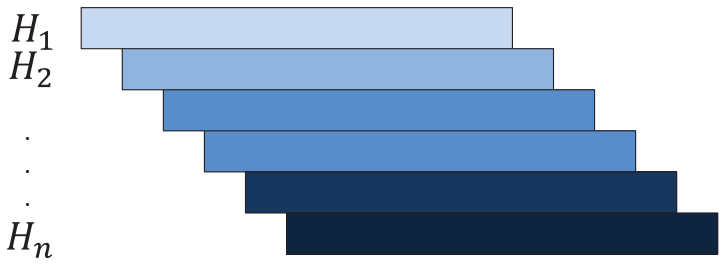


Figure 2.3: Homography-mixture model is consisted of multiple homographies defined over blocks of scanlines. To avoid discontinuities across scanlines, Neighboring homographies are well regularized during the model estimation.

Homography Mixture

A single 2D linear transformation is not enough to describe the non-linear motions between rolling shutter neighboring frames. Grundmann et al.[38] proposed a homography-mixture model to handle the non-linear transformations. As shown in Figure 2.3, Homography-mixture consisted of multiple sub-homographies.

To avoid discontinuities across blocks, homographies are smoothly interpolated by neighboring homographies using Gaussian weights.

$$H_x := \sum_{k=1}^{m} H_k w_k(x). \tag{2.10}$$

where $w_k(x)$ is a gaussian weight centered around the middle of each block k.

Estimation $(x, y) = ([x_1, x_2, 1]^T, [y_1, y_2, 1]^T)$, a pair of matched feature points have the following relation after homography transformation.

$$0 = y \otimes H_x x = y \otimes \sum_{k=1}^{m} H_k w_k(x) x = \sum_{k=1}^{m} w_k(x) \cdot y \otimes H_k x \tag{2.11}$$

where $\otimes$ denotes the cross product. The equation can be rewritten as a set of 2 linear independent equations:

$$A_x^k h_k := \begin{pmatrix} 0^T & -x^T & y_2 x^T \\ x^T & 0^T & -y_1 x^T \end{pmatrix} h_k \tag{2.12}$$

where h_k is the vector formed by concatenating the columns of H_k. Combining all k homographies yields a $2 \times 9k$ linear constraint

$$(w_1(x)A_x^1 ... w_k(x)A_x^k) \begin{pmatrix} h_1 \\ . \\ h_k \end{pmatrix} = A_x h = 0 \tag{2.13}$$

Aggregating all feature matches yields a linear system. After obtaining the homography-mixture for every frame, the path planning stratagem is similar to L_1-based method by replacing a single homography path with multiple-homographies paths.

Chapter 3

Video Stabilization by a Depth Camera

3.1 Introduction

Existing video stabilization methods are limited by two key issues. First, methods relied on homography based frame registration such as [50, 64], suffer from image distortion when there are significant depth changes in a scene. In principle, a homography can register two frames only when the scene is flat, or when there is no camera translation at all. These two conditions are not precisely true in most real videos, and can cause serious distortions in the stabilized results, especially when the distance between scene objects and camera is small such as indoor scenes. Second, long feature tracks are difficult to obtain in scenes with severe occlusion, sudden camera rotation, motion blur, or textureless objects (e.g. white walls in indoor scenes). Hence, methods requiring feature tracking such as [56, 57] tend to fail in these challenging cases.

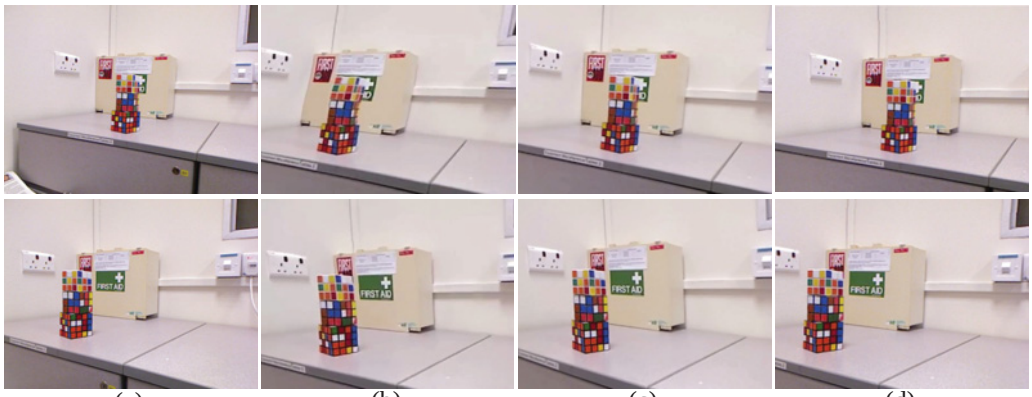
We propose to address these two challenging problems using additional depth sen-

sors. Depth sensors such as the Kinect camera are cheap, compact and widely available in the market. This additional depth information provides video stabilization with a much robust solution in camera motion estimation and frame warping. Since we have depth information, we can estimate an accurate camera pose for each frame by only performing motion estimation between every two consecutive frames. Thus, our method does not rely on fragile feature tracking, or structure-from-motion algorithms [41]. According to our knowledge, this is the first to exploit depth sensors for video stabilization.

Since the depth measure from sensors (e.g., Kinect) is noisy, incomplete and low resolution at each frame, directly applying depth for stabilization is nontrivial. To achieve this goal, we first combine color and depth images to robustly compute 3D camera motion. We match corresponding 2D feature points between two neighboring frames, and use their depths to estimate relative camera motion. We then smooth the recovered 3D camera trajectories following cinematography principles [32], which removes both high frequency camera jitters and low frequency shakes. Since the depth measure is incomplete, the novel video frames cannot be generated by directly projecting 3D scene points (generated from the depth image) according to the new camera poses. To solve this problem, we generate a dense nonlinear motion field by combing 3D projection and 2D image warping to create the final results.

3.2 Indoor Challenge Cases

Before going to the details of our method, we first highlight two key challenges to previous video stabilization methods, which commonly exist in indoor scenes. Indoor scenes are particularly important, because many amateur videos (such as family event,



(a) (b) (c) (d)

Figure 3.1: Results of Cube example. From top to bottom are two sample frames from (a) original video,(b) 2D stabilization method [39], (c) 3D stabilization method [57] and (d) our approach. We can notice clear shear and wobble distortions at the first-aid box and cubes on results (b) and (c), compared with our results.

party, shopping, etc) are captured in indoors. Many of previous methods employed 2D transformations such as similarity [50], affine or homography [64],[39] transformations to register neighboring frames. However, these simple motion models are invalid when there are large depth changes in the scene, especially when the scene is close to the camera. Figure 5.18 shows such an example where three cubes in front of a wall are captured by a handheld video camera. The first row shows two frames of the original shaky video. The second row are the corresponding frames from the video stabilized according to [39]. To produce the results for comparison, we uploaded our videos to Youtube (http://www.youtube.com) with the stabilize feature enabled. The uploaded videos are stabilized by the website server according to the method in [64]. We then downloaded the results for comparison. The results from youtube are clearly distorted. For example, the first-aid box on the left image is subject to a shearing mapping. This is because the sudden depth change between the cubes and the wall makes homography based registration invalid. For a comparison, the same frames from the

video stabilized by our method are shown in the last row. Our method is free from this distortion by exploiting rough depth information from a depth camera.

3D video stabilization methods such as [15, 56, 91] require feature correspondence in different frames for robust 3D reconstruction. Methods based on feature track smoothing such as [50, 57]also need long tracks of feature points. As commented in [57], typically, features should be tracked for about 50 frames to make their algorithm robust. However, robust tracking of feature points is a difficult problem, which could be affected by textureless regions, sudden camera rotation or severe occlusion. The third row of Figure 5.18 shows the results from [57]. To produce the results, we used the stabilize motion with subspace warp in the Adobe After Effects CS5.5 with default parameters (50% smoothness and Rolling shutter automatic reducing) to generate results of [57]. Most of the tracked feature points locate on the foreground cubes, which leads to wobble artifacts on the background first-aid box.

To further demonstrate the tracking difficulty, we show two typical amateur videos in Figure 3.2. Each row shows two frames from one video. The video in the first row has quick rotation, while the one in the second row suffers from severe occlusion caused by pedestrians.We overlay the trajectories of tracked feature points. Here we used the KLT tracker [63] to trace detected SURF features [7]. On each trajectory, the red points are the feature positions in tracked frames. When rotation or occlusion happens, both the number of tracked feature points and the length of feature tracks drop significantly, which makes feature tracking based video stabilization fragile. The average lengthes of feature tracks in the left two images are 10 and 23 frames. In comparison, the average lengthes in the right are 6 and 2 frames. The numbers of tracked points are also reduced from 248 and 158 on the left to 21 and 37 on the right. With an additional depth camera, we compute camera motion between any two

Figure 3.2: Feature point tracking in amateur videos is difficult. Each row shows two frames in a video with quick rotation (top row) or severe occlusion (bottom row). Both the number of tracked points and the length of the feature tracks drop significantly.

consecutive frames from corresponding pixels with known depth. This method does not require long feature tracks. Hence, we avoid this challenging tracking problem.

3.3 Our Method

The input to our method is a video with an accompany depth image for each frame. In developing our algorithm, we use the Kinect camera in indoor scenes for data capturing, though other depth sensors might also be used. Similar to most of the video stabilization methods, our method includes mainly three steps. We first estimate the 3D camera motion from neighboring color and depth images. Since we have depth

information, we do not require long feature tracks for 3D reconstruction. Once the 3D camera trajectory is known, we smooth it following [39] to reduce both high frequency jitters and low frequency shakes. We then generate video frames according to the smoothed camera poses, again by combing information from color and depth images.

3.3.1 Camera Motion Estimation

We begin by recovering camera motion in the original shaky video. Our input are the video frames I_1, I_2, $\cdots$, I_n. and their corresponding depth images P_1, P_2, $\cdots$, P_n. measured in local camera coordinate system. We seek to estimate a 4×4 matrix C_t at each time t that represents the camera pose in a global coordinate system, i.e.

$$C_t = \begin{pmatrix} R_t & O_t \\ 0 & 1 \end{pmatrix}$$

Here, R_t and O_t are the 3×3 rotation matrix and 3×1 translation vectors representing the camera orientation and position in the global coordinate system respectively.

As shown in Figure 3.3, the relative camera motion at time t can be represented by a 3D Euclidean transformation H_t satisfying $C_t = C_{t-1}H_t$. H_t has similar form as C_t, where

$$H_t = \begin{pmatrix} \hat{R}_t & \hat{O}_t \\ 0 & 1 \end{pmatrix}$$

Here, $\hat{R}_t$, $\hat{O}_t$ are the rotation and translation components of H_t. We set the world coordinate system at the first frame. Hence, camera poses can be computed by chaining the relative motions between consecutive frames as $C_t = H_1H_2...H_t$.

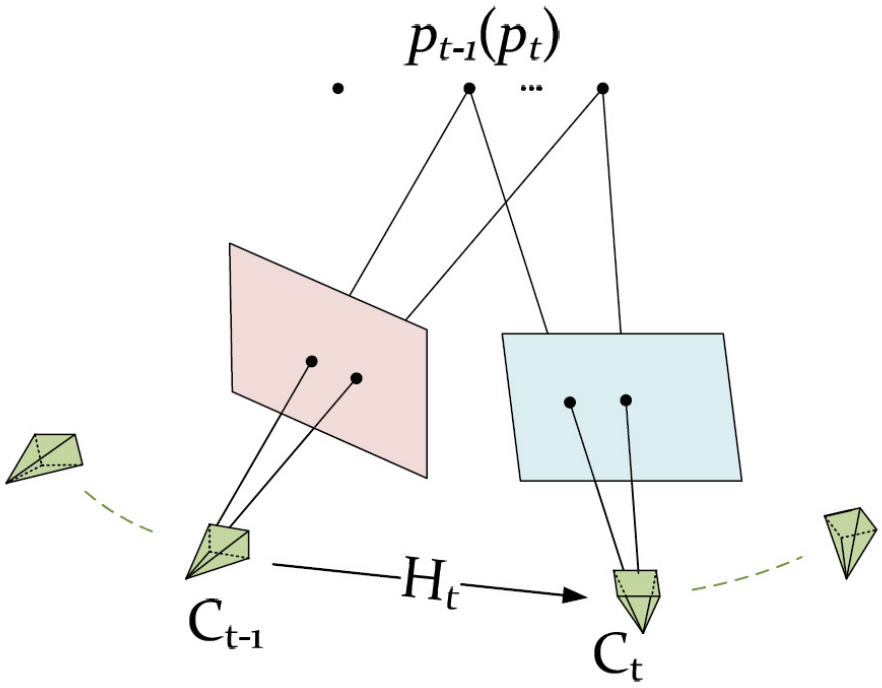


Figure 3.3: Camera motion estimation from corresponding 3D points between two consecutive frames. p_t and p_{t-1} are coordinates of the same 3D point in two local camera coordinate systems. The Euclidean transformation H_t between two cameras can be estimated from corresponding 3D points.

To estimate H_t, we first detect and match SURF features [7] between two frames I_{t-1} and I_t. Since depth images are incomplete (shown on the grayscale image in Figure 5.5(a)), some matched feature points might not have depth recorded. Here, we only choose those corresponding feature points whose depths in both Pt-1 and Pt are known. Each pair of correspondence introduces a constraint about H_t as, $\hat{R}_t p_{t-1} + \hat{O}_t = p_t$. As illustrated in Figure 3.3, p_t, p_{t-1} are the coordinates of the same 3D point in the two local camera coordinate systems of the frame t and t - 1 respectively.

Suppose N pairs of features are collected, we can then estimate H_t (i.e. $\hat{R}_t$, $\hat{O}_t$) by

minimizing

$$\sum_{i=1}^{N} \rho(\|\hat{R}_t p_{t-1} + \hat{O}_t - p_t\|_2). \tag{3.1}$$

Here,$\rho(\cdot)$ is the M-estimator (we use the Tukey bi-weight function [95]) for robust estimation defined as

$$\rho(x) = \begin{cases} \beta^2/6(1-[1-(x/\beta)^2]^3) & if|x| \leq \beta \\ \beta^2/6 & otherwise. \end{cases}$$

Equation 3.1 is minimized by the standard iteratively reweighted least squares (IRLS) method [95]. During the computation, RANSAC is also applied to skip outliers. Specifically, we repetitively draw three randompairs of corresponding points at a time to solve Equation 1 until we find the largest set of inliers. We then solve Equation 3.1 again with all inliers to decide the camera motion. For computation efficiency, during the random sampling, we set $\beta = +\infty$ (i.e. without using M-estimator), while we set β as the standard deviation of the fitting residual in all inliers in the final estimation.

3.3.2 Camera Trajectory Smoothing

We smooth the estimated camera trajectory for stable motion. We follow [39] to adopt cinematography principles to remove both high frequency jitters and low frequency shakes. The smoothed camera trajectory should be a combination of constant, linear and parabolic motion. Note that the key difference from [39] is that we work with real 3D camera poses (i.e. orientations and positions), while [39] used a series of homogrpahies to indicate the camera motion.

We represent the camera rotationmatrix R_t by its quaternions, which offer a better representation for interpolation than Eulerian angles. For notation simplicity, we still

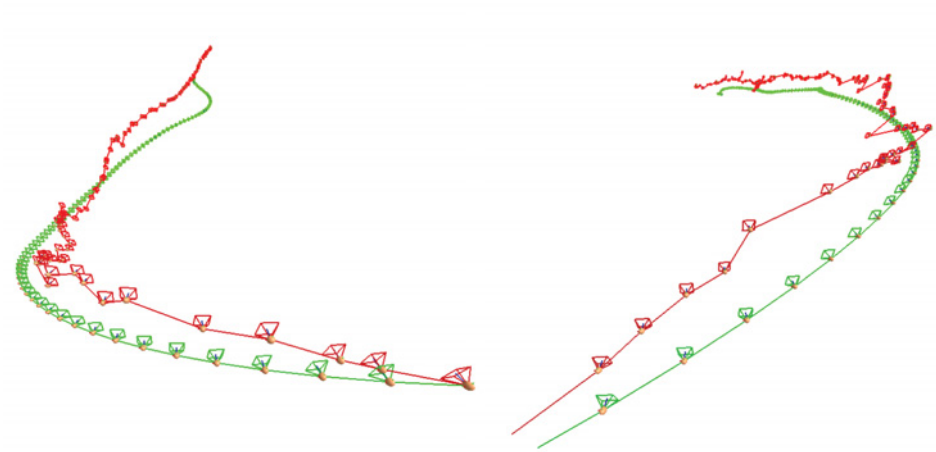


Figure 3.4: Camera trajectory smoothing results. The red and green curves show trajectories before and after smoothing respectively.

denote these quaternions by R_t. We then concatenate the 4D quaternions Rt and the 3D translation vector O_t to a 7D vector F_t to represent the camera pose at time t. The optimal camera trajectory is obtained by minimizing the following objective function,

$$E(F) = w_1|D(F_1)|_1 + w_2|D^2(F)|_1 + w_3|D^3(F)|_1$$

where $|D(F)|_1$,$w_2|D^2(F)|_1$,$w_3|D^3(F)|_1$ are the L-1 norms of the first order, second order and third order camera pose derivatives respectively. We set $w_1 = 10$,$w_2 = 1$,$w_3 = 100$ for all our examples. The optimization is solved by linear programming with the first camera pose F1 unchanged. Following [39], we also require new camera poses to be close to the original ones. Specifically we require the angles in R_t do not change more than 3 degrees and the components in O_t do not change more than 20(20mm). Figure 3.4 shows the camera trajectories before and after smoothing in red and green respectively.

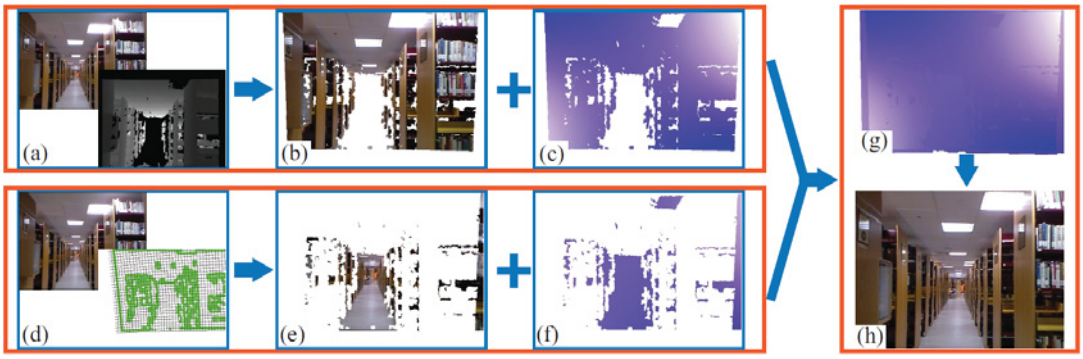


Figure 3.5: Video frame generation pipeline. We use the color and depth images in (a) to generate the projection (b) and the motion field (c). Many pixels are missing because of the incomplete depth image. Hence, we warp the color image by the Content-preserving warp [9] in (d) according to the green control points and a regular grid. This warping generate a color image (e) and a motion field (f). We then generate a complete motion field (g) by fusing (c) and (f). The final video frame (h) is created by warping the original frame with (g).

3.3.3 Video Frame Generation

Once we obtain the stabilized camera poses, we are ready to synthesize the output video. In principle, if the depth sensor returns a dense and complete depth for each pixel, we can generate the stabilized frame by simply projecting all 3D points according to smoothed camera poses. However, the depth image is often incomplete, as shown by the grayscale images in Figure 5.5 (a). Figure 5.5 5 (b) shows a projection of the 3D points (generated from the color and depth image in Figure 5.5) to the stabilized video frame, where many pixels are missing because of the incomplete depth map. Hence, we apply the Content-preserving image warping [56] to fill-in these missing regions.

To seamlessly blend results from projecting 3D points and image warping, we use morphological dilation operator to create a r-pixel width (r = 1.5% of image width in our experiments) buffer band surrounding all missing regions. We use all pixels in this

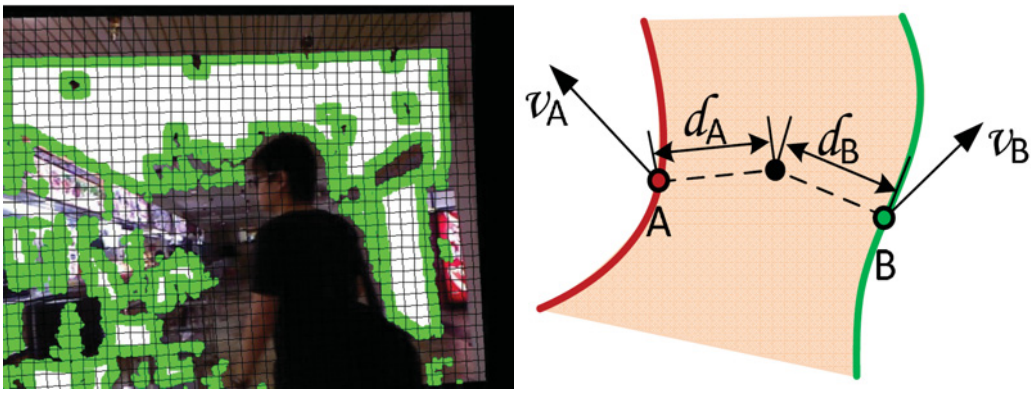


Figure 3.6: Left: control points and image grid for content preserving warp. Right: illustration for motion interpolation.

band as control points for image warping, so that the warping will be as consistent as possible with the projection. Figure 5.5 (d) shows the green control points and the image warping grid (a clearer version is provided in the left of Figure 3.6). We combine these two methods in the band by linearly interpolating the two motion fields introduced by them.

Motion field from depth images. We project pixels with depth measure according to the smoothed camera pose. Given the original camera pose C_t and its smoothed pose C_t' , we can compute the image coordinates of a 3D point p in both original and stabilized video frames. The difference between these two coordinates gives a motion vector, which maps a pixel from original video to the stabilized one. Specifically, the motion vector v for a 3D point p is obtained by: $v = KR_t[I|O_t]p - KR_t'[I|O_t']p$, where K is the camera intrinsic matrix, R_t,O_t and R_t',O_t' are the original and smoothed camera orientation and position respectively. In this way, we obtain a motion field M_t^1 that covers all pixels with depth measure, as shown in Figure 5.5 (c).

Motion field from image warping. To fill in missing regions, we take all pixels in the buffer band as control points for the content-preserving warp. Basically, we partition the original image into 10×10 regular grid. Here we adopt the same energy equation $E = E_d + \alpha E_s$ described in [56], where α is the relative weight of data term E_d and smoothness term E_s. We set $\alpha = 1$ in our implementation. The data term Ed comes from the projected 3D points, we only choose the points located on the boundary of the missing region (green lines of Figure 5.5 (d) and Figure 3.6 on the left). Smoothness term Es controls the rigidity of the gird. The energy equation can be minimized by solving a sparse linear system. After we get the motion of the grid vertices, the motion of a pixel is then computed by bilinear interpolation of the motion vectors at its four grid vertices. This generates another motion field M_t^2, which covers all pixels without depth measure and the buffer band as show in Figure 5.5 (f).

Motion fields blending. We then linearly blend M_t^1 and M_t^2 in the buffer band. Specifically, the motion of a pixel in the band is computed by linearly interpolating the motion of its two nearest neighbors at the two sides of the band. As shown on the right of Figure 3.6, A,B are two pixels on the two sides of the band with minimum distance (d_A, d_B respectively) to the black pixel in consideration. v_A, v_B are the motion vectors of A and B, which are computed from projecting 3D points and image warping respectively. We linearly interpolate these two vectors in the band to blend M_t^1 and M_t^2. For example, the motion of the black pixel is computed as

$$v_B \cdot d_A/(d_B + d_A) + v_A \cdot d_B/(d_A + d_B)$$

Figure 5.5 (g) shows the interpolated motion from (c) and (f). Once the motion field is obtained for the whole frame, we use it to warp the original video frame to create the stabilized frame as shown in Figure 5.5 (h).

3.4 Experiments

We evaluated our method with some challenging videos captured by a Kinect camera. To avoid the calibration between the color and depth cameras, we used the embedded color camera in Kinect whose calibration is known.[1] All our videos have resolution of 640×480. Figure 5.18 and Figure 3.10 compare our results with two state-of-artmethods described in [39] and [57]. In both figures, from the top to the bottom, the four rows for each example are sample frames of the original video, stabilized video according to [39], [57] and our method respectively. For easy reference, we name these examples in Figure 5.18 and Figure 3.10 as Cube and Boy. The Cube and Boy examples showed a nearby scene with sudden depth change, which made the homography based frame registration in [39] fail. Hence, severe geometric distortions were observed in these results (please notice the shear distortion on the first-aid box in the Cube example, and on the bookshelf in the Boy example). The content-preserving warp in [57] is more robust to depth changes. However, the large textureless wall in the Cube example had few tracked feature points, which caused wobble effect in the result. (Note that tracked feature points were used as control points for warping in [57]. Similar artifacts were reported in [56] when the image feature points distributed unequally over the image.) Though more feature points can be tracked in the Boy example, it was not stabilized well by [57], perhaps because the dynamic scene confused

[1] We use OpenNI SDK for our implementation

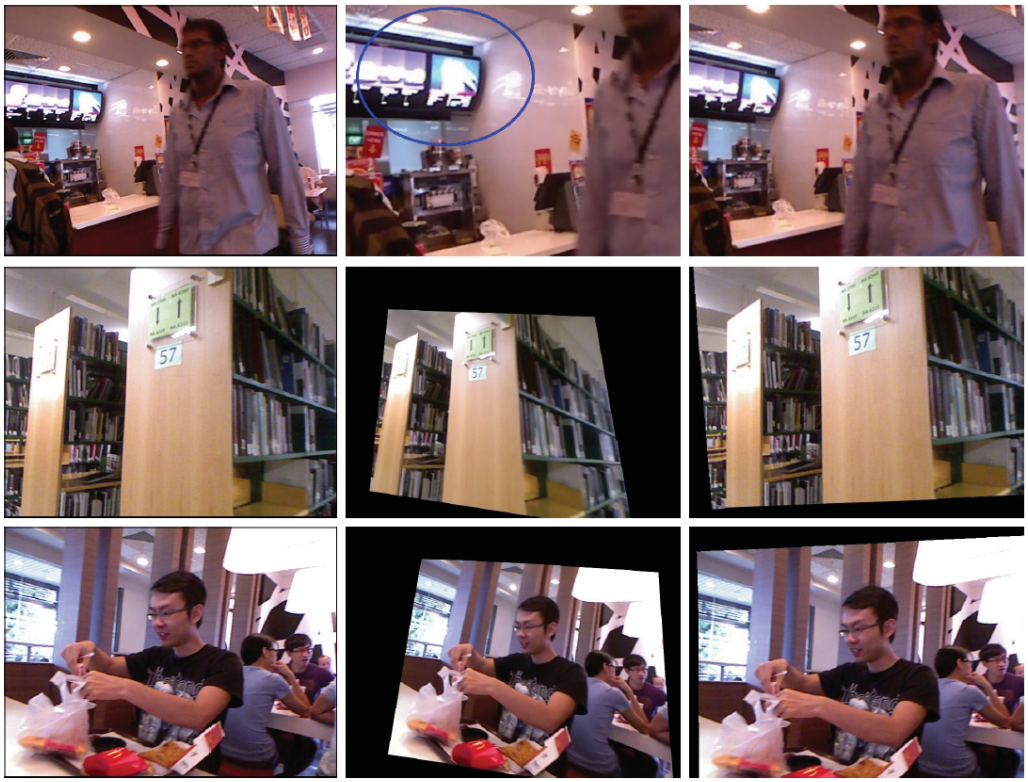


Figure 3.7: Comparison with [57]. Each row shows one example. Columns from left to right: (a)sample frames from original video, (b)stabilized video according to [57] and (c)our method. Please notice the sudden zooming artifacts circled in blue of the first example and warping distortion of the other two examples in (b).

the subspace analysis. In comparison, our method took advantage of the depth and generated better results on these examples.

Figure 3.7 providesmore comparisonwith 3D stabilization method [57]. The first example of Figure 7 contains severe occlusion, where people walked through and blocked the whole frame. It is challenging for [57] because of tracking failures caused by severe occlusion. The region circled in blue had inconsistent motion in the stabilized video (Please refer to our project website). The second and third example of Figure 3.7 contain quick camera rotation. This causes shear artifacts on the whole

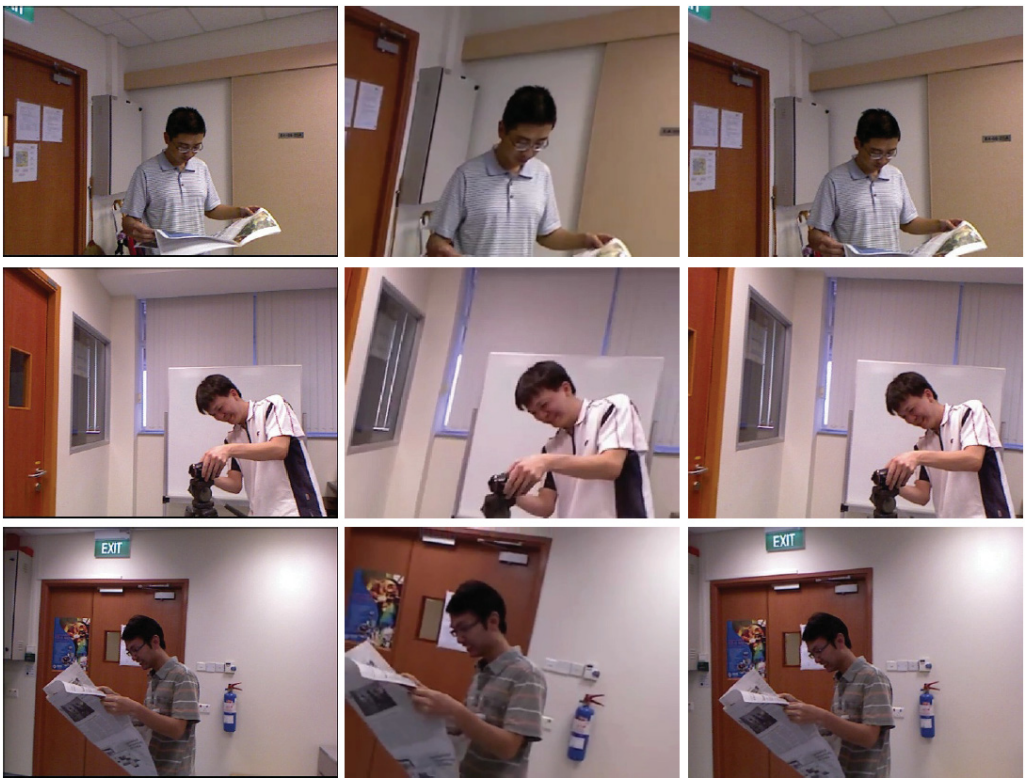


Figure 3.8: Comparison with [39]. Each row shows one example. Columns from left to right: (a) sample frames from original video, (b) stabilized video according to [39] and (c) our method. Please notice the wobble on the background in (b).

scene. Furthermore the warping distortion produces a large empty area. Figure 3.8 shows three examples with severe geometric distortion produced by method [39]. The depth change makes the homography based registration fail. The simple linear model cannot describe variations of depth in these scenario. Please notice the shear distortion on the background in Figure 3.8 (b).

Limitations We observe several limitations of our approach, which point out the direction for future study. First, our method does not consider the rolling shutter effects

Figure 3.9: Additional results under different indoor environment from our video stabilization, which are shown in the project page.

of both the color camera and the depth camera, which sometimes make the camera motion estimation imprecise and lead to some high frequency jitters in the results. Second, our current implementation is limited to the Kinect camera, which only works in indoor scenes. But we believe the same algorithm can be also applied to time-of-flight cameras in outdoor environments.

3.5 Conclusion

We studied two challenges in video stabilization, namely sudden depth change which makes 2D motion model imprecise and tracking failure which causes 3D stabilization fail. We solved these problems with an additional depth sensor, which provides a depth measure for each video frame. We exploited this rough depth information to improve both camera motion estimation and frame warping. Our results demonstrated the effectiveness of the proposed method.

Figure 3.10: Results on the Boy examples. From top to bottom, the four rows are sample frames from (a) original video, (b) stabilized video according to [39], (c) stabilized video according to [57] and (d) our method.

Chapter 4

Bundled Camera Paths for Video Stabilization

4.1 Introduction

A video captured with a hand-held device (e.g., a cell-phone or a portable camcorder) often appears remarkably shaky and undirected. In the previous chapter, we introduced a stabilization method captured by a depth sensor. It is of great practical importance to focus on the traditional devices such as mobile phones, tablets and camcorders.

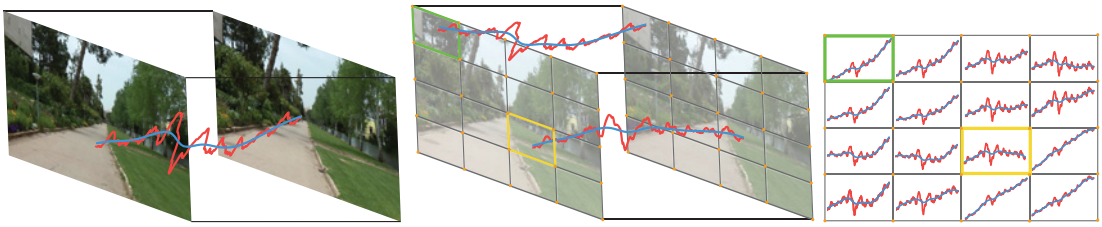
Prior video stabilization methods synthesized a new stabilized video by estimating and smoothing 2D camera motion or 3D camera motion. In general, 2D methods are more robust and faster because they only estimate a linear transformation (affine or homography) between consecutive frames. But the 2D linear motion model is too weak to fundamentally handle the parallax caused by non-trivial depth variation in the scene. On the contrary, the 3D methods can deal with the parallax in principle and generate strongly stabilized results. However, their motion model estimation is less

robust to various degenerations such as feature tracking failure, motion blur, camera zooming, and rapid rotation. Briefly, 2D methods are more robust but may sacrifice quality (e.g., introducing unpleasant geometrical distortion or producing less stabilized output), while 3D methods can achieve high-quality results but are more fragile.

This work aims at the same goal of robust high-quality result but from an opposite direction: we propose a more powerful 2D camera motion model. Specifically, we present bundled camera paths model which maintains multiple, spatially-variant camera paths. In other words, each different location in the video has its own camera path. This flexible model allows us to fundamentally deal with nonlinear motion caused by parallax and rolling shutter effects. At the same time, the model enjoys the robustness and simplicity of 2D methods, because it only requires feature correspondences between two consecutive frames.

Our bundled camera paths model is built on two novel components: a warping-based motion representation (and estimation), and an adaptive space-time path smoothing. The first component represents the motion between two consecutive frames by mesh-based, spatially-variant homographies (Figure 4.1 1(b)) with a ss-similar-as-possible regularization constraint [42, 71]. This constraint is critical because estimating a model with such a high degree of freedom is usually risky in the cases of insufficient features or large occlusions. To the best of our knowledge, this is the first work to employ the mesh-based as-similar-as-possible regularization for spatially-variant motion estimation in video stabilization. Notice that the as-similar-as-possible warping was used in [56, 57] for video stabilization. But we directly use the mesh vertices as the motion model itself. No intermediate representation is used, such as 3D reconstruction [56] et al. or subspace [57].

Based on the proposed motion representation, we construct a bundle of camera



(a) a single global path (b) our bundled paths

Figure 4.1: Comparison between traditional 2D stabilization (a single global camera path) and our bundled camera paths stabilization. We plot the camera trajectories (visualized by the y-axis translation over time) and show the original path (red) and the smoothed path (blue) for both methods. Our bundled paths rely on a 2D mesh-based motion representation, and are smoothed in space-time.

paths, each of which is the concatenation of local homographies at the same grid cell over time (Figure 4.1 1(b)). Our second component smooths all bundled camera paths as a whole to maintain both spatial and temporal coherences. Furthermore, to avoid excessive cropping/geometrical distortion and approximate cinematography favored path, we adopt a discontinuity-preserving idea similar to bilateral filtering [82] to adaptively control the strength of smoothing.

For a quantitative evaluation, we provide a comprehensive dataset (including both public examples and our own video clips of different kinds of motions). We show that our new 2D method is comparable to or outperforms other competitive 2D or 3D methods.

4.2 Bundled Camera Paths

In this section, we introduce our warping-based motion model and bundled camera paths.

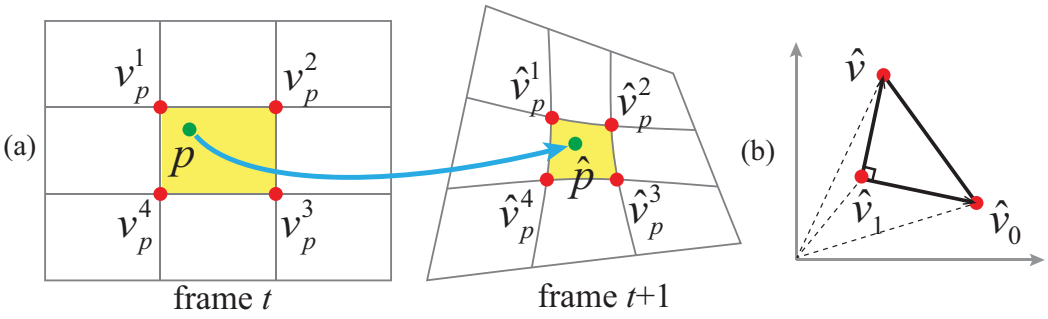


Figure 4.2: (a) Parameterization of the motion between two frames by a regular grid mesh, where a pair of matched features $(p, \hat{p})$ should be represented by the same bilinear interpolation of their four enclosing vertices. (b) The as-similar-as-possible term requires each triangle $\hat{v}, \hat{v}_0, \hat{v}_1$ to follow a similarity transformation.

4.2.1 Warping-based Motion Model

We propose using an image warping model to represent the motion between consecutive video frames, which provides stronger modeling power than conventional single, 2D linear transformations. We adopt the warping model in [42, 56], though more general models such as 'moving-least-square' [71] or parameterized optical flow [69] might be used.

Model At each frame, we define a uniform grid mesh as illustrated in Figure 4.2. The motion is represented by an (unknown) warping of the grid mesh to register two frames (in fact, their corresponding feature points). We require matched features (*e.g.*., p and $\hat{p}$ in Figure 4.2) to share the same bilinear interpolation of the four corners of the enclosing grid cell after warping. At the i-th grid cell, the warping from frame t to frame $t+1$ introduces a homography $F_i(t)$, which can be determined from the motion of the four enclosing vertices. Thus, the warping-based motion model is actually a set of spatially-variant homographies on a 2D grid.

Note that this highly flexible model is able to handle parallax. It is between global

homography and per-pixel optical flow. However, estimating a model with such a high degree of freedom is very risky because we may not have sufficient features (due to textureless regions or occlusions) in every cell.

Regularization To address this challenge, we propose imposing a shape-preserving (*i.e.*., "as-similar-as-possible" [42]) constraint. The combination of the shape-preserving and mesh representation together provides two kinds of regularizations: 1) for each cell, the fitted homography should be biased toward a reduced similarity (or rigid) transformation; 2) the intrinsic connection of the mesh (two neighboring mesh cells share two vertices) enforces a first-order continuity constraint. They can help to propagate or fill in information from regions with sufficient features to other regions.

Finally, we estimate the motion by minimizing two energy terms: a data term for matching features, and a shape-preserving term for enforcing regularization.

4.2.2 Model Estimation

We first describe our basic method by following [56], and later extend it for better robustness in the next subsection.

Data term As shown in Figure 4.2, suppose $\{p, \hat{p}\}$ is the p-th matched feature pair from frame t to frame $t+1$. The feature p can be represented by a 2D bilinear interpolation of the four vertices $V_p = [v_p^1, v_p^2, v_p^3, v_p^4]$ of the enclosing grid cell: $p = V_p w_p$, where $w_p = [w_p^1, w_p^2, w_p^3, w_p^4]^\top$ are interpolation weights that sum to 1. We expect that the corresponding feature $\hat{p}$ can be represented by the same weights of the warped grid vertices $\hat{V}_p = [\hat{v}_p^1, \hat{v}_p^2, \hat{v}_p^3, \hat{v}_p^4]$. Therefore the data term is defined as

$$E_d(\hat{V}) = \sum\nolimits_p ||\hat{V}_p w_p - \hat{p}||^2. \tag{4.1}$$

with shape-preserving no shape-preserving

Figure 4.3: Comparison of motion estimation with and without the shape-preserving term.

Here $\hat{V}$ contains all the warped grid vertices. Solving $\hat{V}$ determines the warping of the grid.

Shape-preserving term We use the same shape-preserving term as [56] involving all vertices in $\hat{V}$,

$$E_s(\hat{V}) = \sum_{\hat{v}} \|\hat{v} - \hat{v}_1 - sR_{90}(\hat{v}_0 - \hat{v}_1)\|^2, \; R_{90} = \begin{bmatrix} 0 & 1 \\ -1 & 0 \end{bmatrix}, \tag{4.2}$$

where $s = \|v - v_1\|/\|v_0 - v_1\|$ is a known scalar computed from the initial mesh. This shape-preserving term requires the triangle of neighboring vertices v, v_0, v_1 to follow a similarity transformation. Linearly combining two terms forms our final energy $E(\hat{V})$:

$$E(\hat{V}) = E_d(\hat{V}) + \alpha E_s(\hat{V}), \tag{4.3}$$

where α is an important weight to control the amount of regularization. We will discuss how to adaptively determine it later. Since the energy $E(\hat{V})$ is quadratic, the warped mesh $\hat{V}$ can be easily solved by a sparse linear system solver.

Estimating homographies After having a new mesh, we can estimate each local ho-

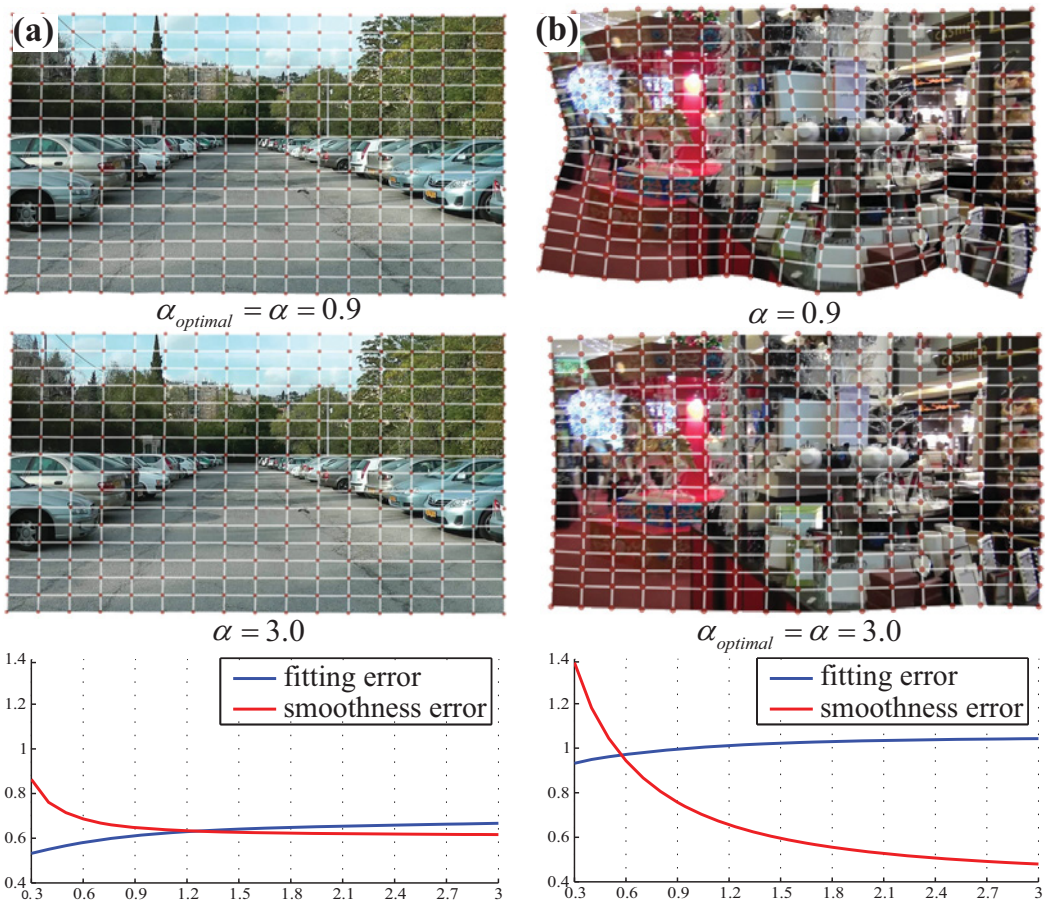


Figure 4.4: Our method automatically chooses an appropriate α for different scenes: (a) a scene free of occlusion; (b) a scene with severe occlusion.

mography $F_i(t)$ in the grid cell i of frame t by solving a linear equation:

$$\hat{V}_i = F_i(t)V_i, \tag{4.4}$$

where V_i and $\hat{V}_i$ are the four vertices before and after the warping.

Figure 4.3 shows the warped mesh grid according to the estimated motion. Left and right are the results with and without the shape-preserving term. It is clear that the regularization term helps maintain a smooth varying mesh representation.

4.2.3 Robust Estimation

We further generalize our motion estimation to make it more robust.

Outlier rejection We reject incorrectly matched features at two scales. At the coarse scale (the whole image), we apply RANSAC algorithm [24] to fit a global homography $\bar{F}(t)$ and discard features by a relatively large threshold on fitting error (6% image width). At the fine scale (4×4 sub-images), we apply RANSAC again to reject features by a relatively small threshold (2% image width).

Pre-warping To facilitate the warping estimation, we use global homography $\bar{F}(t)$ to bring matching features closer. We then solve the warping to estimate the residual motion, which generates a homography $F_i'(t)$ at each grid cell. The final homography $F_i(t)$ is simply computed as $F_i'(t)\times\bar{F}(t)$. Note that this coarse-to-fine strategy has been used in [56] for image synthesis and proven effective in motion estimation literature [11].

Adaptive regularization A good regularization should be adaptive to image content. For example, if reliable features are uniformly distributed over the whole image, we should trust the data term more and use a smaller weight α in Equation (4.3) for a weaker regularization. But when there is occlusion or insufficient features, we prefer stronger regularization as the data term is less reliable. To implement this strategy, we adaptively set α per frame, based on two errors: fitting error e_h and smoothness error e_s.

The fitting error e_h is the average residual of the feature matching under the estimated homographies, i.e., $e_h = \frac{1}{n}\sum_p \|F_p \times p - \hat{p}\|^2$, where F_p is the homography in the cell containing p, and n is the number of feature pairs. The smoothness error e_s measures the similarity (L_2 distance) between neighboring local homographies by

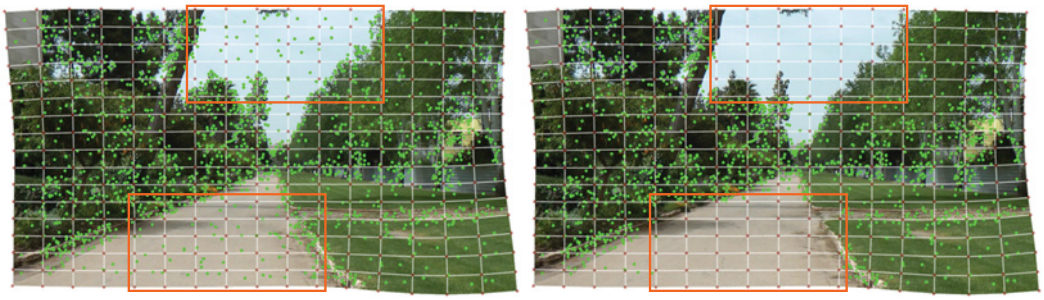


Figure 4.5: Left: the estimated warping mesh from all feature points. Right: we exclude all the features in the orange box when estimating the warping model. A similar mesh can be obtained despite the lack of features.

$e_s = \beta \sum_{j \in \Omega_i} \|F_i - F_j\|^2$, where Ω_i consists of the neighboring cells of i. Here, the homography matrix is normalized so that sum of all its elements is one. We empirically set $\beta = 0.01$, since it makes the scale of e_h and e_s similar on most of the examples. Then we define the combined error as $e = e_h + e_s$. We equally discretize α into 10 values between 0.3 and 3. We perform the model estimation using every discretized value and select the model with minimum error e.

As shown in Figure 4.4(a), for simple scenes with smooth depth variation, neighboring cells tend to have similar homographies. So we choose a small α(=0.9) to better minimize the data error. On the contrary, for scenes with large occlusion (Figure 4.4(b)), neighboring local homographies are less similar. The smoothness error can be significantly reduced by increasing α. So our system will automatically choose a large α(=3.0) to ensure consistent local motion.

Finally, we show an example in Figure 4.5 to verify the strength of the regularization of our method. In this example, we compare two meshes estimated using all features and a subset of features. Two similar results indicate our method can robustly deal with regions of insufficient features.

4.2.4 Bundled Camera Paths

With estimated local homographies, we can define a bundle of spatially-variant camera paths for the whole video. Let $C_i(t)$ be the camera pose of the grid cell i at frame t. It can be written as:

$$C_i(t) = C_i(t-1)F_i(t-1), \Rightarrow C_i(t) = F_i(0)F_i(1)\cdots F_i(t-1),$$

where $\{F_i(0), ..., F_i(t-1)\}$ are estimated local homographies at the same grid cell i, as shown in Figure 4.6 (a). We call these spatially-variant paths as "bundled camera paths". In the next section, we describe how we smoothen these bundled paths for video stabilization.

4.3 Path Optimization

We first describe our smoothing method for a single camera path, and extend it to a bundle of camera paths.

4.3.1 Optimizing a Single Path

A good camera path smoothing should consider multiple competing factors: removing jitters, avoiding excessive cropping, and minimizing various geometrical distortions (shearing/skewing, wobble). To reach a desired balance, we propose an optimization-based framework taking all factors into account.

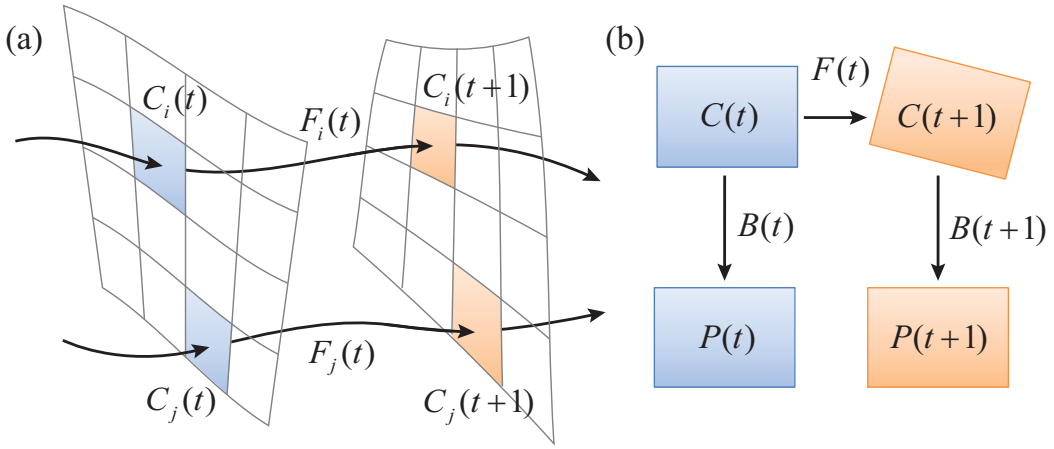


Figure 4.6: (a) Bundled camera paths. (b) Relationships among original path $\{C(t)\}$, smoothed path $\{P(t)\}$, and transformations $\{B(t)\}$

Formulation Given an original path $\mathbf{C} = \{C(t)\}$, we seek an optimized path $\mathbf{P} = \{P(t)\}$ by minimizing the following function:

$$\begin{aligned} &\mathcal{O}\left(\{P(t)\}\right) = \\ &\sum_t \left(\|P(t) - C(t)\|^2 + \lambda_t \textstyle\sum_{r \in \Omega_t} \omega_{t,r}\left(\mathbf{C}\right) \cdot \|P(t) - P(r)\|^2 \right), \end{aligned} \tag{4.5}$$

where Ω_t are the neighborhood at frame t. The other terms are:

- data term $\|P(t) - C(t)\|^2$ enforcing the new camera path to be close to the original one to reduce cropping and distortion;
- smoothness term $\|P(t) - P(r)\|^2$ stabilizing the path;
- weight $\omega_{t,r}\left(\mathbf{C}\right)$ to preserve motion discontinuities under fast panning/rotation or scene transition;
- parameter λ_t to balance the above two terms.

Since Equation (4.5) is quadratic, we can solve it with any linear system solver. Here, we use a Jacobi-based iterative solver [9]:

$$P^{(\xi+1)}(t) = \frac{1}{\gamma}C(t) + \sum_{r\in\Omega_t, r\neq t} \frac{2\lambda_t\omega_{t,r}}{\gamma} P^{(\xi)}(r), \tag{4.6}$$

where $\gamma = 1+2\lambda_t\sum_{r\in\Omega_t, r\neq t}\omega_{t,r}$, and ξ is an iteration index. At initialization, $P^{(0)}(t) = C(t)$. Once we obtain the optimized path $\mathbf{P}$, we compute the warping transform $B(t) = C^{-1}(t)P(t)$ to warp the original video frame to the stabilized result (Figure 4.6(b)).

Discontinuity-preserving The adaptive weight $\omega_{t,r}$ is important to preserve motion discontinuity. We follow the idea of bilateral filter [82] and design it by two Gaussian functions:

$$\omega_{t,r} = G_t\left(\|r-t\|\right) \cdot G_m\left(\|C(r)-C(t)\|\right), \tag{4.7}$$

where $G_t()$ gives larger weight to the nearby frames. $G_m()$ measures the changes of two camera poses.

We use a large kernel to ensure successful suppression of both high-frequency jitters (*e.g.*, handshake) and low-frequency bounces (*e.g.*, walking). In our implementation, we set Ω_t to 60 neighboring frames and the standard deviation of $G_t()$ to 10. In contrast, previous low-pass filtering based methods [64] typically need a smaller amount of support (*e.g.*, 10 frames) to avoid aggressive cropping and distortion. But such a small kernel is often insufficient in suppressing low frequency bounces.

The reason why we can use a larger kernel lies in $G_m()$. In video stabilization, for rapid camera motion (e.g, caused by fast panning or scene transition), an inappropriate

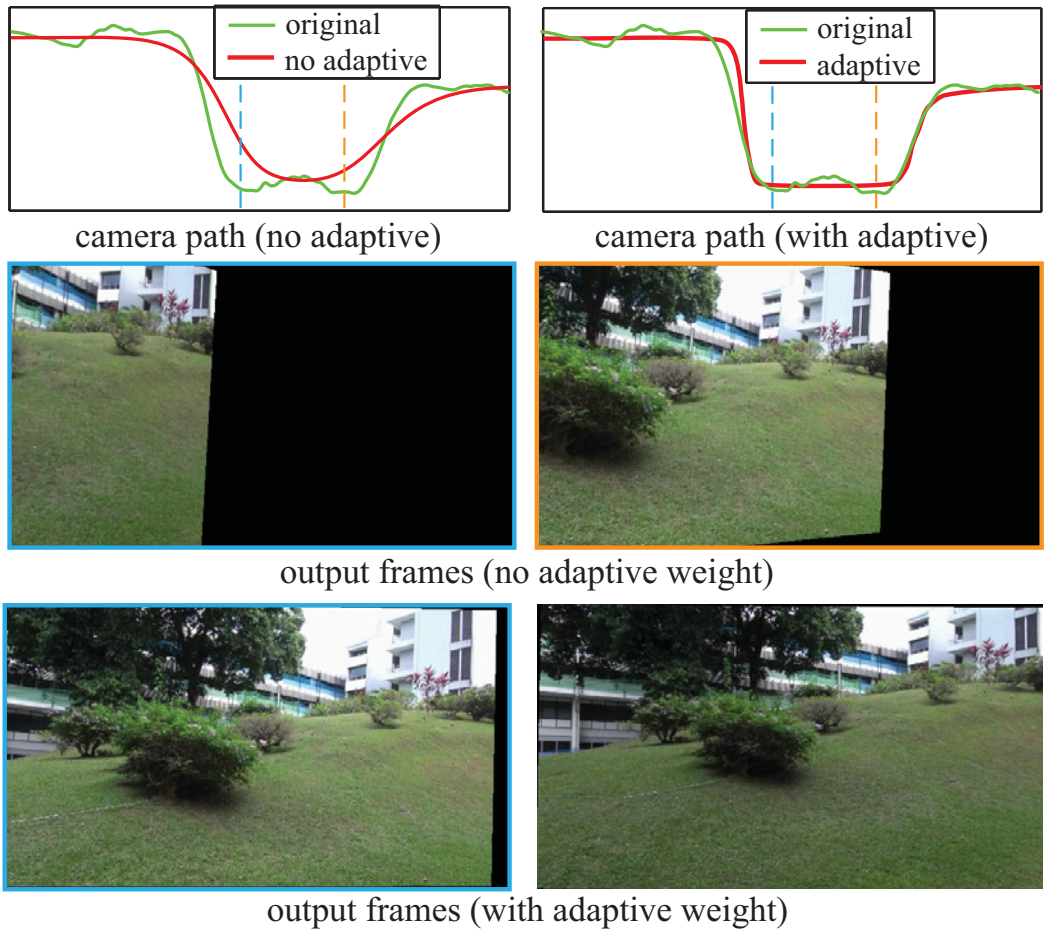


Figure 4.7: Comparison of with and without adaptive weights $G_m()$ for a video with rapid camera panning. The camera paths on the top plot the x-translation over time.

amount of smoothing may lead to excessive cropping, as shown in Figure 4.7. In this case, the camera pans quickly, and naïve Gaussian smoothing (second row) causes the camera path to significantly deviate from its original path, as indicated by the dashed lines in the left plot on top. The corresponding frames shown on the second row will require large cropping. Our adaptive term $G_m()$ preserves the sudden camera motions to a certain degree. The result from our adaptive smoothing (bottom row) produces much less cropping.

To measure the camera motion, we use the change in translation components $\mu_x(t)$, $\mu_y(t)$ extracted from the camera pose $C(t)$, namely $|\mu_x(t) - \mu_x(r)| + |\mu_y(t) - \mu_y(r)|$. The frame translation $\mu_x(t)$, $\mu_y(t)$ can describe most camera motions in practice except for an in-plane rotation or scale around the principal axis.

Cropping and distortion control The above adaptive term $\omega_{t,r}$ can give us a certain amount of ability to control cropping and distortion. However, the user may want to have strict control on the cropping ratio and distortion. In principle, we could formulate a constrained optimization to address this issue. But it may be too complex to be solved or reproduced.

In this work, we resort to a simple but effective method - adaptively adjust the parameter λ_t for each frame. We first run the optimization with a global fixed $\lambda_t = \lambda$ (empirically set to 5) and then check the cropping ratio and distortion of every frame. For any frame that does not satisfy the user requirements (cropping ratio or distortion is smaller than a pre-defined threshold), we decrease its parameter λ_t by a step $(1/10\lambda_t)$ and re-run the optimization. Note, according to Equation 4.6, a smaller λ will make the optimized path closer to the original one, which has less cropping and distortions. The procedure is iterated until all frames satisfy the requirements.

We measure the cropping ratio and distortion from the warping transform $B(t) = C^{-1}(t)P(t)$. The anisotropic scaling of $B(t)$ measures the distortion. It can be computed by the ratio of the two largest eigenvalues of the affine part of $B(t)$ [41]. We use $B(t)$ to compute the overlapping area of the original video frame and the stabilized frame. The cropping ratio is the ratio of this area and the original frame area. In our experiments, we require the cropping ratio to be larger than 0.8, and the distortion score to be larger than 0.95 for all examples. In principle, we can further measure the perspective distortion by the two perspective components in $B(t)$. But we empirically find they are always too small when compared with the affine components and do not include them.

4.3.2 Optimizing Bundled Paths

Our motion model generates a bundle of camera paths. If these paths are optimized independently, neighboring paths could be less consistent, which may generate distortion in the final rendered video. Hence, we do a space-time optimization of all paths by minimizing the following objective function

$$\sum_i \mathcal{O}\left(\{P_i(t)\}\right) + \sum_t \sum_{j \in N(i)} \|P_i(t) - P_j(t)\|^2, \tag{4.8}$$

where $N(i)$ includes eight neighbors of the grid cell i.

The first term is the objective function in Equation (4.5) for each single path, and the second term enforces the smoothness between neighboring paths. This optimization is also quadratic and the optimum result can be obtained by solving a large sparse

linear system. Again, our solution is updated by a Jacobi-based iteration [9]:

$$P_i^{(\xi+1)}(t) = \frac{1}{\gamma'}(C_i(t) + \sum_{\substack{r\in\Omega_t \\ r\neq t}} 2\lambda_t w_{t,r} P_i^{(\xi)}(r) + \sum_{\substack{j\in N(i) \\ j\neq i}} 2P_j^{(\xi)}(t)),$$

where

$$\gamma' = 2\lambda_t \sum\nolimits_{r\in\Omega_t, r\neq t} w_{t,r} + 2N(i) - 1.$$

We typically iterate 20 times to optimize camera paths.

During optimization, the motion-adaptive term $G_m(\cdot)$ is evaluated at individual cells, since different cells have different motion. In comparison, λ_t is determined from the global path (generated by concatenating the pre-warping global homographies), because it controls the overall cropping and distortion. Then, we use λ_t to optimize the camera paths in all cells.

Result synthesis After path optimization, we compute the warping matrix $B_i(t)$ for each cell i by $B_i(t) = C_i^{-1}(t)P_i(t)$. We then apply $B_i(t)$ to warp the i-th cell at the t-th frame to generate the final output video. Usually, applying $B_i(t)$ directly generates good results. This is because our motion estimation ensures first order smoothness of the original paths. Furthermore, the bundled optimization in Equation (4.8) requires nearby optimized paths to be similar. Thus, the smoothness is naturally satisfied by $B_i(t)$ most of the time. Sometimes, there are slight distortions (*e.g.*., seams of about 1-pixel width), in which case we perform a bilinear interpolation to fix them.

4.3.3 Correcting Rolling Shutter Effects

Our bundled paths model can naturally handle rolling shutter effects without pre-calibration. The principle of our method is similar to that of [38]. Our system does rolling shutter correction while simultaneously stabilizing the video. In a shaky video, a rolling shutter causes spatially variant high frequency jitters. When smoothing the camera paths, we simultaneously rectify rolling shutter effects and other jitters caused by camera shake.

4.4 Results

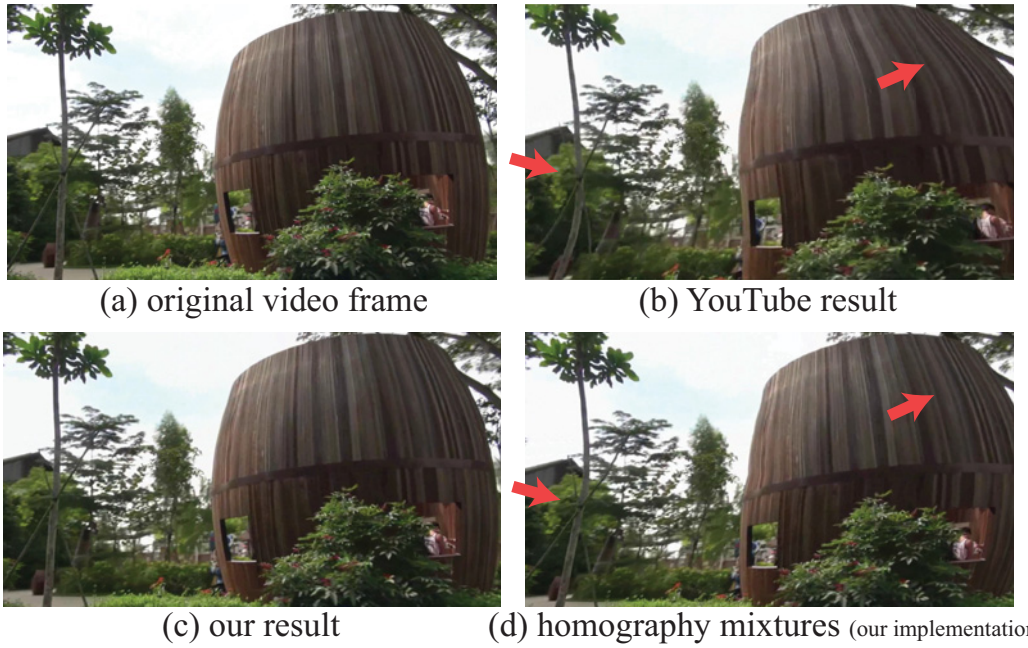
We run our method on an Intel i7 3.2GHZ Quad-Core machine with 8G RAM. We extract 400-600 SURF features [7] per frame. For motion estimation, we always divide the video frame to 16×16 cells. For a video of 1280×720 resolution, our un-optimized system takes 392 milliseconds to process a frame (around 2.5fps). Specifically, we spend 300ms, 50ms, 12ms and 30ms to extract features, estimate motion, optimize camera paths and render the final result. All original and result videos are provided on our webpage[1].

4.4.1 Algorithm Validation

We first verify the effectiveness of different components of the proposed approach.

A Global Path vs. Bundled Paths For the example in Figure 4.1, the result according to a global path has remaining jitters in some image regions. This is because the parallax makes the global homography motion model invalid, therefore some image

[1] http://www.liushuaicheng.org/SIGGRAPH2013/index.htm



(a) original video frame (b) YouTube result

(c) our result (d) homography mixtures (our implementation)

Figure 4.8: Comparison with the homography mixture models in Grundmann et al. [38]. (a) A sample frame in the original video. (b) The output frame produced by YouTube Stabilizer. (c) The result produced by our method. (d) The result produced using our implementation of homography mixture[38] (with the same bundled path smoothing).

regions cannot be stabilized very well. But our bundled paths can handle this kind of typical situation. Please refer to our accompanying video for a visual comparison.

Spatially-variant Homographies vs. Homography Mixture Grundmann et al. [38] proposed a homography mixture model for rolling shutter correction. They divide a video frame into a 1D array of horizontal blocks, and use a Gaussian mixture of homographies for each block. This model is beyond a single 2D transformation and able to partially handle parallax.

Compared with our 2D mesh-based, spatially-variant homographies, this model has two limitations: 1) it does not address horizontal depth variation; 2) it uses weaker feature points (which apply lower threshold level for feature detection) and a simple

Gaussian mixture for the regularization. Weaker feature points may result in larger fitting errors and the ability to use simple Gaussian smoothing is limited.

Figure 4.8 shows a comparison of these two models. In this example, the scene has horizontal depth variation and the sky region lacks feature points. Figure 4.8 (a) is the result of using YouTube Stabilizer (integrated Homography Mixture feature). We can observe severe geometrical distortions. To further verify our observation, we replace our spatially-variant model with the homography mixture model (our implementation) in our framework and generate the result in Figure 4.8 (d), where we observe similar distortion. In comparison, our warping-based motion estimation can fundamentally handle depth variation (not limited to vertical direction). Our result (Figure 4.8 (c)) does not suffer from such distortion.

Rolling Shutter Handling Figure 5.15 compares our methods with [38] on two example videos from their paper. Our model accounts for frame distortions such as skew (left example) and local wobble (right example). More examples are included in the supplementary video, which shows we achieve similar results on correcting rolling shutter distortion as [38].

4.4.2 Quantitative Evaluation

To quantitatively evaluate and measure the result from different aspects, we define three objective metrics.

Cropping and distortion Our first two metrics measure *cropping ratio* and *global distortion*. We first fit a global homography at each frame between input video and output video. We then compute the cropping ratio and distortion for each frame. The

Figure 4.9: Two rolling shutter removal examples using our method and [Grundmann et al. 2012 [38]]. Our results are on par with that from [Grundmann et al. 2012 [38]].

cropping ratio can be directly computed from the scale component of the homography. There is one global cropping ratio for the whole sequence, and each frame provides an estimation. We average these estimations at all frames as the final metric. The distortion is computed as defined in Section 4.3.1. Because any distortion in a single frame will destroy the perfection of the whole result, we choose their minimum across the whole sequence as the final metric. This "worst-case" metric allows us to easily see whether the whole result video is completely successful. For a good result, both metrics should be close to 1.

Stability The third metric measures the *stability* of the result. Designing a good metric is non-trivial because it is hard to compare two different videos. We suggest an empirically good metric using frequency analysis on estimated 2D motion from a video. Our basic assumption is that the more energy is contained in the low frequency part of the motion, the more stable a video is.

Computationally, we estimate our bundled camera paths to approximate the true motion (optical flow) in a video. We do not smooth out anything after the estimation. Then, we extract translation and rotation components from each path. Each component is a 1D temporal signal. Finally, we evaluate the energy percentage of the low frequency components (expect for DC component) in these 1D signals to measure the stability.

Specifically, we take a few of the lowest (empirically set as from the 2nd to the 6th) frequencies and calculate the energy percentage over full frequencies (excluded by the DC component). Similar to the distortion, we take the smallest measurement among the translation and rotation as the final metric. For a good result, the metric should approach 1 here as well.

4.4.3 Comparison with Publicly Available Results

The purpose of this comparison is to test whether our results are comparable with (if not better than) previous "successful" results in [56, 57, 35, 39]. We collect eleven test videos from these papers (thumbnails in Figure 4.10), and compare our results with their published results (all from authors' project webpages).

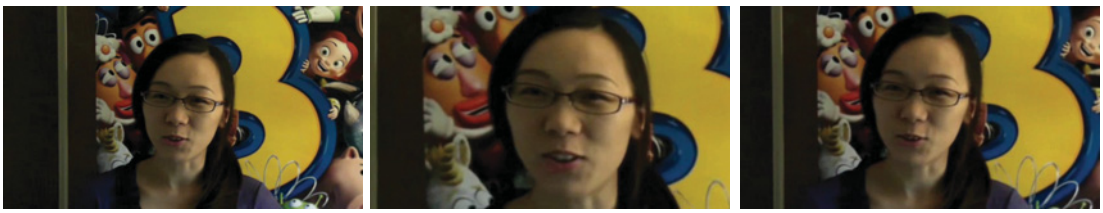
Overall, all methods generate similar stability both subjectively and quantitatively (Figure 4.10) on these examples, while our results are slightly better on some videos in terms of cropping ratio and distortion.

For video (2)-(4), 3D stabilization [56] achieves the best stability and distortion scores. It suggests that 3D methods are the first choice (in term of stability and distortion error), when the 3D motion can be successfully estimated. Although our results are slightly worse in stability, the visual difference is quite small (please verify from the supplementary video). Furthermore, the aggressive smoothing in 3D methods sometimes leads to an output FOV that is too small as demonstrated by the cropping score. Our method manages to provide a good trade-off. For video (5-9), [57], [35], and our method achieve similar stability, while our method is slightly better in cropping and distortion. For video (10-11)[1], our method outperforms the L1-optimization [39] in stability (slightly), cropping ratio, and distortion scores.

Figure 4.11 highlights the most challenging video (10) in this dataset. Liu et al. [57] refer this example as a failure case because a single subspace cannot account for the feature trajectories on both the face and the background. Their results have visible distortion. [39] produced better result on this example. But in the video result, we still observe large temporal distortion on the background region. (See our accompanying

[1]To better measure stability on background motion (caused by camera shake), we use a manual foreground mask to exclude foreground motion.

Figure 4.10: Quantitative comparison with existing stabilization techniques on publicly available data.



input [Liu et al. 2011] our result

Figure 4.11: Comparison with a failure case of prior methods.

video.) In comparison, our method can successfully handle this example (achieve best in terms of all three metrics) because the warping-based motion model can represent this complicated motion.

4.4.4 Comparison with the State-of-the-Art Systems

Due to no publicly available implementation of previous works, we compare our system with two well-known commercial systems – YouTube Stabilizer and 'Warp Stabilizer' in Adobe After Effects CS6. The YouTube Stabilizer is based on the combination of the L_1-norm path optimization [39] and homography mixtures [38]. The 'Warp Sta-

bilizer' in Adobe After Effects is largely based on subspace stabilization [57]. We understand that commercial products are often different from a given research system. But we believe these two systems represent the essential elements of research conducted in this field, and the comparison makes sense for examining strengths or weaknesses and robustness (for various videos using a set of fixed parameters) of our system.

Dataset We assemble a comprehensive dataset of 174 short videos ($10 \sim 60$ seconds) from previous publications, Internet, and our own captures. To know the strength and weakness of a method in different situations, we roughly divide our data into 7 categories based on camera motion and scene type. They are: (I) *simple*, (II) *quick rotation*, (III) *zooming*, (IV) *large parallax*, (V) *driving*, (VI) *crowd*, and (VII) *running*.

YouTube Stabilizer is a parameter-free online tool. But 'Warp Stabilizer' is an interactive system, and the user might carefully tune a few parameters. Here, we wish to examine its robustness as an automatic tool by fixing its parameters. We use the example videos in [57] to decide the best parameters. Finally, we choose the default parameters (smoothness: 50%, 'Smooth Motion' and 'Subspace Warp') to produce results.

Quantitative Comparison For each category, we compute the average metrics and standard deviation of three systems (Figure 4.12 (a)). We discuss the results with regard to each system in detail below.

All three systems perform well in category (I) "*simple*", since this category contains videos with relatively smooth camera motion and mild depth variations. Though our method has a minor advantage, the users can safely choose any of three to get a desired

result.

Among the remaining categories, we want to highlight the category (IV) "*large parallax*". The three systems achieve similar stability, while our system is clearly better in terms of distortion. We show two examples in Figure 4.12 (b) and (c) for visual comparison of our system and the YouTube Stabilizer. These examples show the limitation of a 1D array of homography mixtures – it cannot model depth changes in horizontal direction. Warp Stabilizer also generates some shearing/skewing artifacts in some video frames, though in principle this 3D method should be able to handle parallax. Figure 4.12 (d) shows such an example (please note the shearing of the bookshelf). This is probably due to the subspace analysis failure caused by occlusion. Our method succeeds in all of these examples. Comparison in this category clearly demonstrates the advantages of our warping-based motion model in dealing with a large parallax.

Categories (II–III) contain quick rotation or zooming, which are challenging cases for methods requiring long feature tracking. 'Warp Stabilizer' often generates significant cropping. Figure 4.12(e) is such an example. To alleviate this problem, we try to interactively tune its smoothing parameters. When applying a weaker smoothing, however, we find its result becomes shaky. In comparison, our method generates stable results with much less cropping. For categories (V–VII), the three systems generate similar stability levels ('Warp Stabilizer' is slightly better in category VII), while our system is consistently better with respect to either cropping ratio or distortion control.

We notice that our method generates relatively smaller standard deviations of the three metrics for all categories. It suggests that our method generates more consistent results from various inputs.

User Study We further conduct a user study with 40 participants to evaluate and compare our method with the YouTube Stabilizer and the 'Warp Stabilizer' in Adobe AfterEffects CS6. Every participant is required to evaluate results on 28 different input videos (randomly sampled from our dataset), in which there are 4 videos for each category mentioned above (The 4 video are prepared in the way that two of them compare our result to YouTube Stabilizer, and the other two to 'Warp Stabilizer'). In the user study, we use the scheme of forced two-alternative choice. Every participant is asked to pick a better one between the results of our method and YouTube Stabilizer, or between the results of our method and the 'Warp Stabilizer'. These videos are displayed to the subjects in a random order. The subjects are unaware of the video categories. Neither do they know which technique is used to produce the stabilized results. Figure 4.13 (a) shows such an interface for the user study. The original video is displayed on the top. The two stabilized ones are shown side-by-side below. Users can simultaneously play input video and both two results to better examine the difference. And these videos can be played back and forth, or be paused at a certain frame to help users carefully make their decision. The user can also play each of these videos individually to examine their quality without other distractions. We ask users to disregard differences in aspect ratio, or sharpness since each one may undergo different video codecs or further post-processing which makes uniform treatment difficult.

The user study results are shown in Figure 4.13 (b). For each category, we show the average percentage of user preference. In general, the majority of all users showed significant preference towards our results when compared to any of the other two systems respectively. In particular, the participants prefer the overall quality of our results for category (IV) "*large parallax*" over YouTube Stabilizer (72% vs. 28%) and 'Warp Stabilizer' (69% vs. 31%). The result is consistent with our metric evaluation. For

category (II–III) containing quick rotation or zooming, users show a strong bias in preference toward our results over 'Warp Stabilizer' (93% vs. 7% for rotation, 83% vs. 17% for zooming). This is possibly due to the significant cropping in the results of 'Warp Stabilizer'. For categories (V–VII), more participants prefer our results to the other two systems, although the three systems generate similar stability levels according to our stability metric. It is likely because of the superior distortion and cropping control in our method. In category (I) "*simple*", users express similar preference toward three results.

After the user study, we also ask all participants to articulate the criteria for their feedbacks. We conclude the main criteria for unacceptable videos: 1) the video gets a smaller field of view or even contains frames with visible empty (black) area; 2) the video presents structure distortions in individual frames; 3) the motions in some video frames vibrate or oscillate; 4) the scene transition looks abrupt or not smoothed in the video. From these criteria, our proposed metrics can be partially related with human preferences. And both quantitative evaluation and user study results consistently indicate our system performs better than the other two systems.

4.4.5 Limitations and Discussion

We find that when 3D reconstruction is successful, 3D methods often generate the best results. However, our system is more robust as we do not require feature tracking, and it produces comparable or only slightly worse results. It is interesting to note that our adaptive path optimization can also be applied to path smoothing for 3D methods [56, 57, 35], which often use low-pass filtering (Gaussian smoothing), or curve fitting for path planning. In comparison, our adaptive camera path smoothing tech-

nique can automatically adjust the smoothness strength by considering discontinuity and distortion. We show such an example video on our project webpage.

There are cases where the warping-based motion model fails to handle severe occlusions or dis-occlusions, especially when combined with rolling shutter effects. Our warping-based motion model chooses a large α to enforce strong coherence between grid cells. In this way, we can minimize the geometrical distortion, but at the same time, we sacrifice motion accuracy and eventually the stability of the result. In general, we find geometrical distortion is more disruptive than some slight remaining jitters.

Our path optimization does not strictly follow cinematography rules, which may be desirable in certain applications. But our discontinuity-preservation optimization produces visually pleasing results in most examples. If necessary, we could apply the strategy in [32] as a post-process to solve this problem. We also do not deal with motion blur. Sometimes, the stabilized results contain visible blur artifacts. This problem can be addressed by the recent work [20].

4.5 Conclusion

We have presented a new 2D video stabilization method with a bundled camera paths model. The proposed method can simultaneously generate comparable results to 3D methods while keeping merits of 2D methods. Using image warping techniques for motion representation is an interesting finding. In the future, we would extend this kind of representation to other video-based applications.

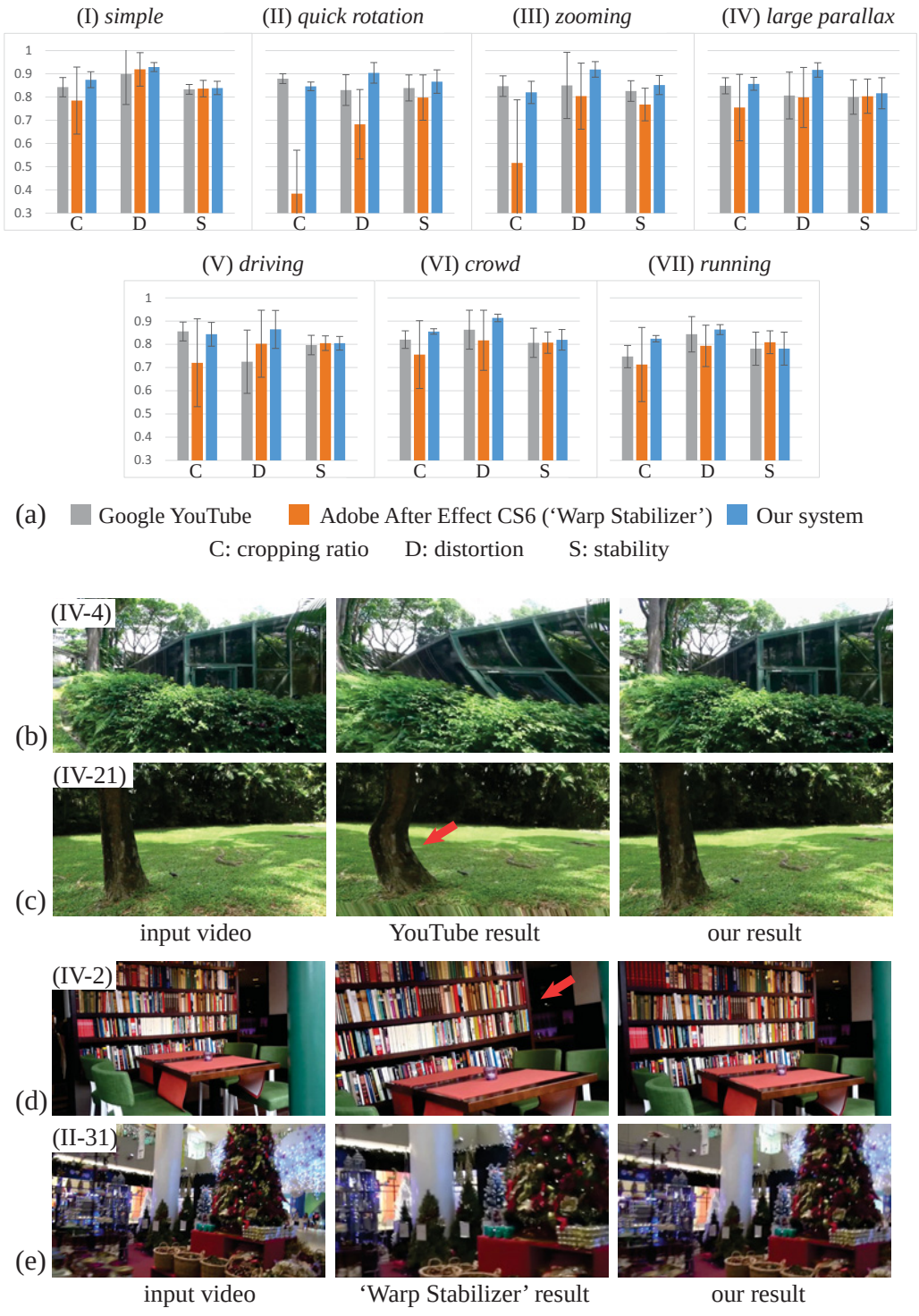


Figure 4.12: Comparisons with two popular systems: YouTube Stabilizer and Adobe After Effect "Warp Stabilizer". Top: quantitative comparisons by three metrics: cropping (C), distortion (D) and stability (S). Bottom: some sample video frames for visual comparisons.

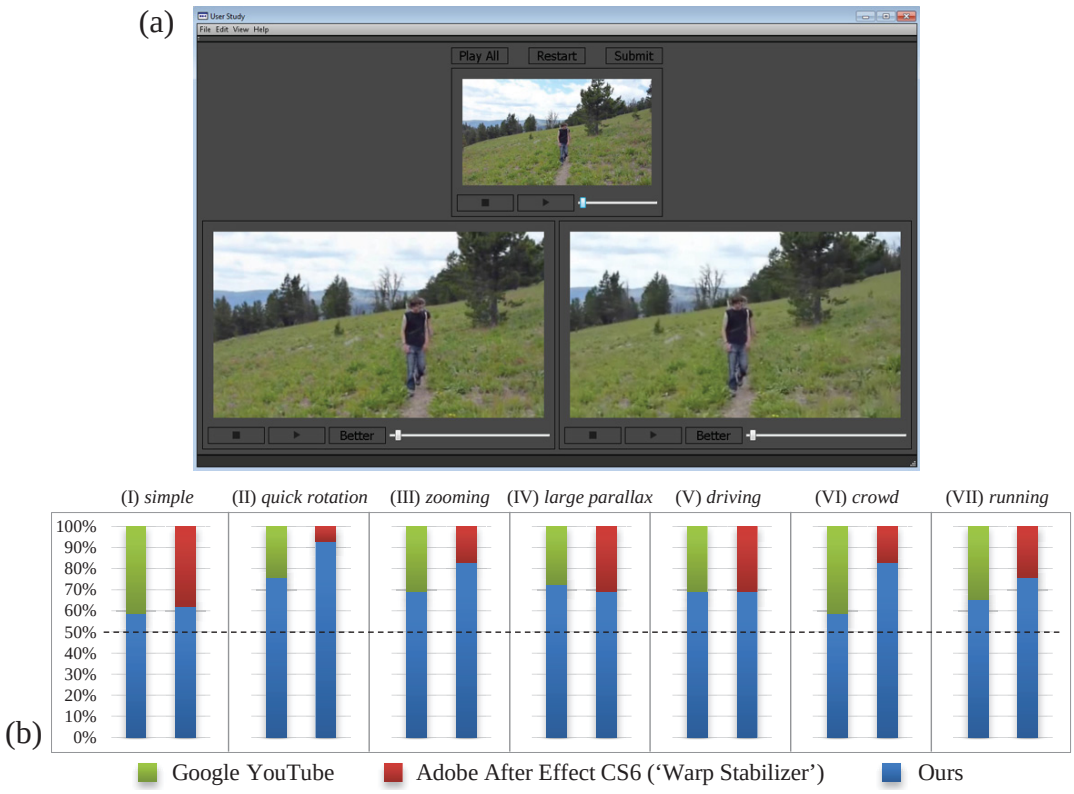


Figure 4.13: (a) Pair-wise comparison interface for user study. (b) User study results by comparing our method with two popular systems: YouTube Stabilizer and Adobe After Effect "Warp Stabilizer".

Chapter 5

SteadyFlow: Spatially Smooth Optical Flow for Video Stabilization

5.1 Introduction

Video stabilization results heavily rely on the adopted motion model. Some methods assume a parametric 2D motion model (such as homography [65] or a mixture of homography [38, 61]) between consecutive frames. These methods are robust but have limited power to deal with spatially variant motion. Feature trajectories provide more flexible non-parametric 2D motion representation. Some recent methods [57, 35]achieve good stabilization results by smoothing feature trajectories. However, dealing with feature trajectories is complicated. Feature trajectories are often spatially sparse and unevenly distributed. They might end or start at any frame of the video. Furthermore, obtaining long trajectories is hard in consumer videos (e.g.,due to rapid camera panning or motion blur).

Dense 2D motion field (such as optical flow) is a more flexible and powerful motion

model. When the optical flow is spatially smooth, we find that smoothing feature trajectories can be well approximated by smoothing pixel profiles, which are motion vectors collected at the same pixel location over time. In other words, we can smooth pixel profiles instead of smoothing feature trajectories. No feature tracking is required. All the pixel profiles begin at the first frame and end at the last frame. This is a much desired property for robust stabilization of consumer videos. The optical flow of a general video could be rather discontinuous, especially on moving objects and strong depth edges. Therefore, we require to modify the raw optical flow to get a SteadyFlow. The SteadyFlow is a close approximation of the optical flow, by enforcing strong spatial smoothness, so that we can simply smooth the pixel profiles extracted from the SteadyFlow to stabilize the original video. We initialize the SteadyFlow by traditional optical flow and identify discontinuous motion vectors by a spatial-temporal analysis. These discontinuous flows are removed and missing regions are filled in by motion completion. We evaluate our method on different types of challenging videos. The experiment results demonstrate the robustness of our technique.

5.2 SteadyFlow Model

In this section, we introduce the concept of pixel profiles. We will further explain why a shaky video can be stabilized by directly smoothing the pixel profiles of the SteadyFlow. Then we demonstrate the SteadyFlow model and the advantages over feature trajectories.

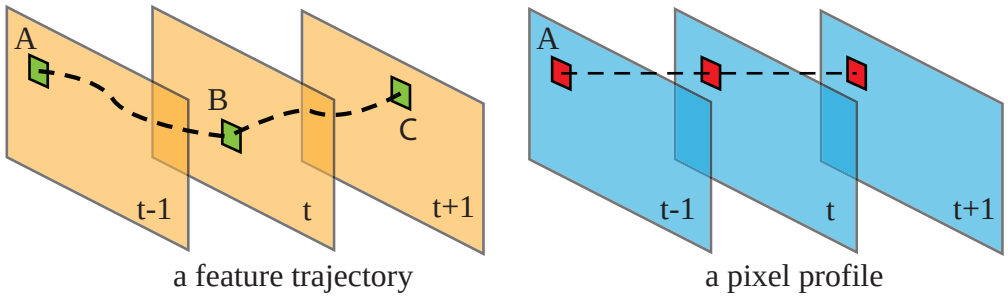


Figure 5.1: Feature trajectory vs. pixel profile. A feature trajectory tracks a scene point while a pixel profile collects motion vectors at the same pixel location.

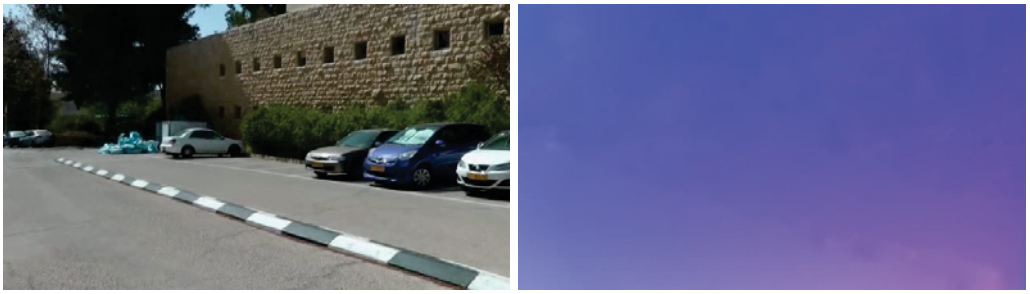


Figure 5.2: A simple static scene (from [35]) with gradual depth variations and its optical flow. This video can be stabilized by smoothing all the pixel profiles extracted from its optical flow.

5.2.1 Pixel Profiles vs. Feature Trajectories

A *pixel profile* consists of motion vectors collected at the same *pixel location*. In comparison, a feature trajectory follows the motion of a scene point. Figure 5.1 shows a feature trajectory and a pixel profile starting at the pixel A in frame $t-1$. The feature trajectory follows the movement from pixel A in frame $t-1$ to pixel B in frame t, and then to pixel C in frame $t+1$. In comparison, the pixel profile collects motions at a fixed pixel location A over time.

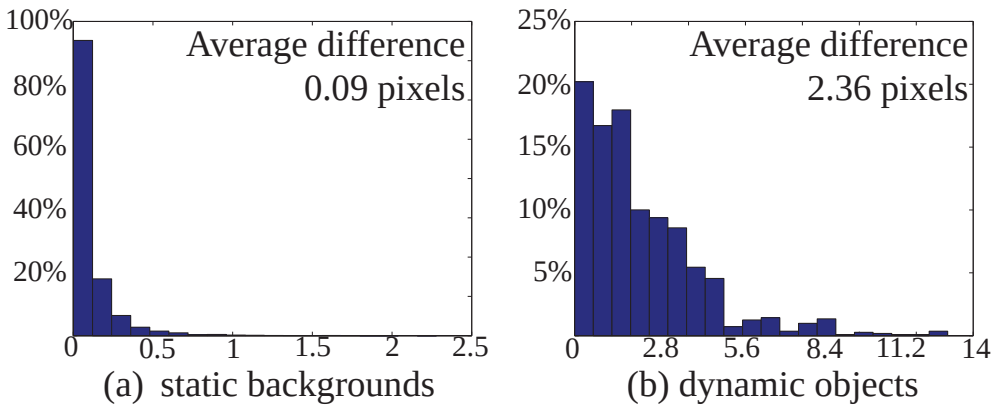


Figure 5.3: Histogram of the difference between feature trajectories and pixel profiles on static backgrounds and dynamic objects.

5.2.2 Stabilization by Smoothing Pixel Profiles

We begin with a simple example. Figure 5.2 shows an video of static scene with gradual depth changes. Its optical flow is spatially smooth as shown on the right side. We simply smooth all the pixel profiles extracted at every pixel location (the technique of smoothing will be presented in Section 5.4). In this way, we can obtain a well stabled output video. This suggests that a video can be stabilized by smoothing pixel profiles.

To understand that, we examine 108 videos in a publicly available dataset[1]. We compute optical flows between all consecutive frames on these videos. We also run a KLT tracker[63] to all videos to get feature trajectories. We further manually mark out moving objects in all video frames assisted by Adobe After Effect CS6 Roto brush. In this way, we collect 14,662 trajectories on static backgrounds and 5,595 trajectories on dynamic objects with the length no less than 60 frames. We compare the difference between a feature trajectory and the pixel profile which begins from the starting point

[1] http://www.liushuaicheng.org/SIGGRAPH2013/index.htm

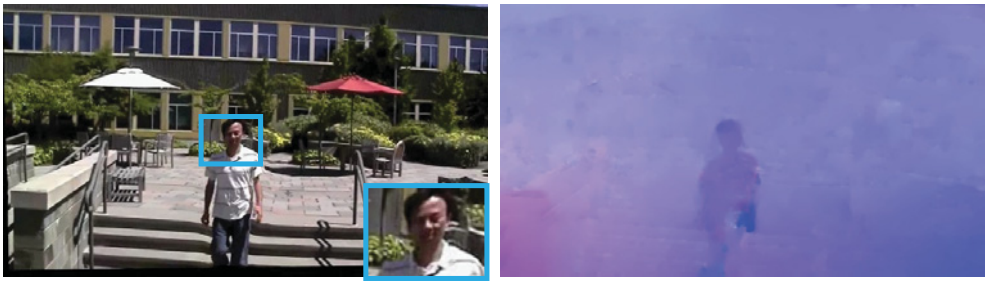
of the trajectory. The difference is evaluated as the average of all motion vector differences between the feature trajectory and the pixel profile at corresponding frames. The histogram of this difference for all trajectories is shown in Figure 5.3. In Figure 5.3(a), we can see over 90% of feature trajectories on static backgrounds are very similar to their corresponding pixel profiles (less than 0.1-pixel motion difference). This suggests that smoothing the feature trajectories can be well approximated by smoothing the pixel profiles. In comparison, as shown in Figure 5.3 (b), the difference between a feature trajectory and its corresponding pixel profile is large on moving objects.

5.2.3 SteadyFlow

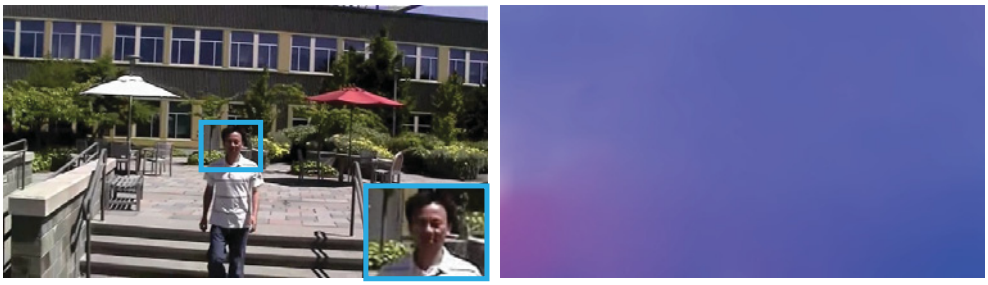
The analysis in Figure 5.3 (b) suggests that pixel profiles can be very different from feature trajectories sometimes. In Figure 5.4 (a) and (c), we show two videos with more complicated optical flow fields to study this problem further. As we can see, the flow vectors are discontinuous on the walking person and strong depth edges. If we smooth the pixel profiles of the raw optical flow, we observe severe image distortions, as illustrated in the close-up views. This indicates that smoothing pixel profile generates poor results on discontinuous flows.

We seek to modify the raw optical flow to get a SteadyFlow. The SteadyFlow should satisfy two properties. First, it should be close to the raw optical flow. Second, it should be spatially smooth to avoid distortions. With these properties, a video can be stabilized by smoothing all its pixel profiles collected from the SteadyFlow. In Figure 5.4 (b) and (d), we show the results by smoothing the pixel profiles generated from the SteadyFlow (shown on the right side). The results are free from artifacts.

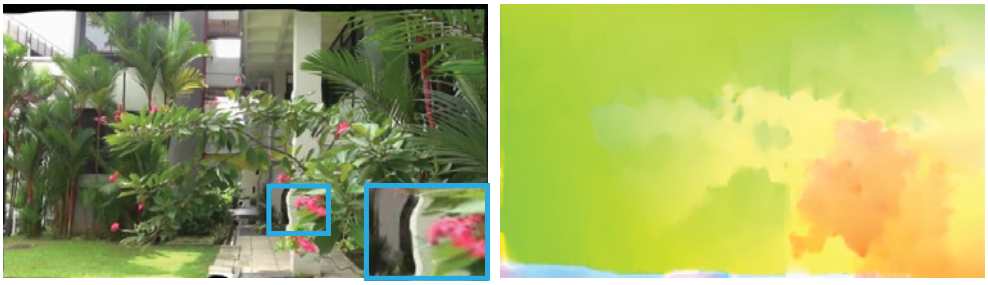
Note that a simple Gaussian smoothing of the raw optical flow is insufficient, as the



(a) smoothing pixel profiles collected from raw optical flow (with dynamic object)



(b) smoothing pixel profiles collected from our SteadyFlow



(c) smoothing pixel profiles collected from raw optical flow (with depth edge)

(d) smoothing pixel profiles collected from our SteadyFlow

Figure 5.4: Comparisons between optical flow and our SteadyFlow. (a) and (c): On the left side, we show the videos stabilized by smoothing the pixel profiles according to the raw optical flow. Please see the distortions highlighted in close-up views. The optical flow field is visualized on the right side. (b) and (d): Corresponding results according to our SteadyFlow.

smoothing will propagate the motions on the moving objects to the background, which decreases the frame registration accuracy and generates temporal wobbles nearby the moving object. Instead, we identify, discard discontinuous flow vectors, and fill in missing flows to satisfy the two desired properties of SteadyFlow. The details will be presented in Section 5.3.2 and Section 5.3.3.

5.2.4 Advantages over Feature Trajectories

In video stabilization, the pixel profiles are superior to the feature trajectories for several reasons. First, the pixel profiles are spatially and temporally dense. In comparison, feature trajectories are sparse, unevenly distributed, and would reach out of the video frame. So it is much harder to design a good filter to smooth feature trajectories. Second, accurate long feature trajectories are difficult to obtain. Though we might get dense feature trajectories by frame-by-frame tracing optical flow, these feature trajectories suffer from significant drifting errors [21]. Third, smoothing feature trajectories independently would introduce severe distortions. Some extra constraints (e.g. subspace projection [57]) are required before smoothing. In comparison, as we will see later, pixel profiles can be smoothed individually as long as the flow field is spatially smooth.

Pixel profiles rely on the quality of optical flows. Optical flow estimation is often imprecise at textureless regions and object boundaries. In most of the time, textureless regions have few structure, so that they introduce little visible distortions. The inaccuracy of flows at object boundaries is largely alleviated by our discontinuity abolition and motion completion.

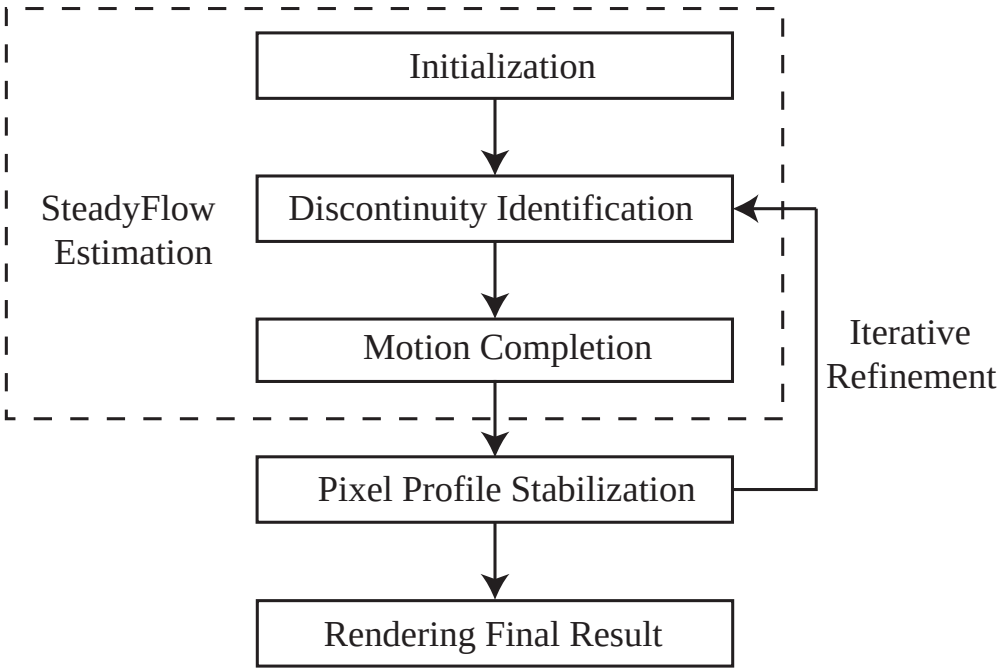


Figure 5.5: Pipeline of SteadyFlow stabilization.

5.3 SteadyFlow Estimation

Our stabilization system pipeline is illustrated in Figure 5.5. We first initialize the SteadyFlow by a robust optical flow estimation. To enforce spatial smoothness, we then identify discontinuous motion vectors and overwrite them by interpolating the motion vectors from neighboring pixels. Then, pixel profiles based stabilization is applied on the SteadyFlow. We adopt an iterative approach to increase the accuracy of SteadyFlow estimation. The final result is rendered according to the stabilized SteadyFlow.

5.3.1 Initialization

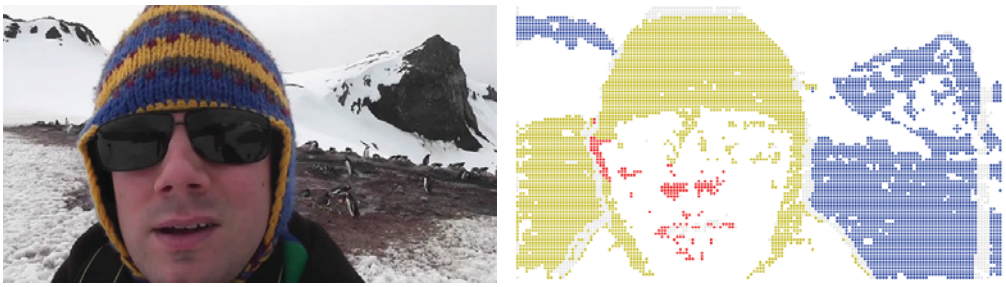
We initialize the SteadyFlow with a robust optical flow estimation. We first estimate a global homography transformation from the matched KLT features [63] between two video frames. We align them accordingly, and then apply the optical flow algorithm described in [53] to compute the residual motion field. The SteadyFlow is initialized as the summation of the residual optical flow and the motion displacements introduced by the global homography.

5.3.2 Discontinuity Identification

A possible solution to detect different motions is to adopt the motion segmentation techniques [74]. However, motion segmentation itself is a difficult problem. Many methods require long feature trajectories. Though there are two-frame-based motion segmentation techniques[23, 68], typically it is still challenging to deal with large foreground objects due to insufficient motion contrast between neighboring frames.

In Figure 5.6, we show the limitation of a recent motion segmentation method [12] on a shaky video. The left-side portions of background features are incorrectly assigned to the foreground face.

We introduce a novel spatial-temporal analysis to identify pixels with discontinuous flow vectors. These pixels are viewed as 'outlier' pixels. We use an outliers mask $\mathbf{M}_t(p)$ to record if pixel p at frame t is 'outlier' (e.g. $\mathbf{M}_t(p) = 0$) or not (ie, $\mathbf{M}_t(p) = 1$). In the spatial domain, we threshold the gradient magnitude of raw optical flow to identify discontinuous regions. Once the magnitude at p is larger than the threshold (0.1 in our experiment), p is considered as 'outlier'. The spatial analysis can only detect boundary pixels on moving objects, because the motion vectors within



(a) input frame (b) motion segmentation result

Figure 5.6: Failure of motion segmentation. (a) a sample input video frame. (b) the motion segmentation results by [12]. Points in the same color are segmented together.

a moving object is often coherent, though they are different from the motions on the background. Therefore, we will further adopt a temporal analysis to identify them.

The temporal analysis examines the accumulated motion vectors $c_t(p) = \sum_t u_t(p)$, where $u_t(p)$ is the motion vector on pixel p at frame t, to decide if p is 'outlier'. It is based on the observation that, in a stable video, the accumulated motion vectors $c_t(p)$ should be smooth over time, except on moving objects and strong depth edges. Figure 5.19 shows a stabilized video and the accumulated motion vectors at two pixel positions. The pixel (marked by a white star) always lies on the static background. Its accumulated motion vectors generate a smooth trajectory over time (shown in Figure 5.19 (b)). In comparison, as shown in Figure 5.19 (a), the trajectory of accumulated motion vectors at the other pixel (marked by a white dot) has significant amount of high frequencies at the beginning, because a moving person passes through that pixel in the first few frames. Its trajectory becomes smooth when the person moves away. We

compute the outlier mask $\mathbf{M}_t(p)$ as:

$$\mathbf{M}_t(p) = \begin{cases} 0, & (\|c_t(p) - \mathcal{G} \otimes c_t(p)\| > \varepsilon) \\ 1, & otherwise. \end{cases} \tag{5.1}$$

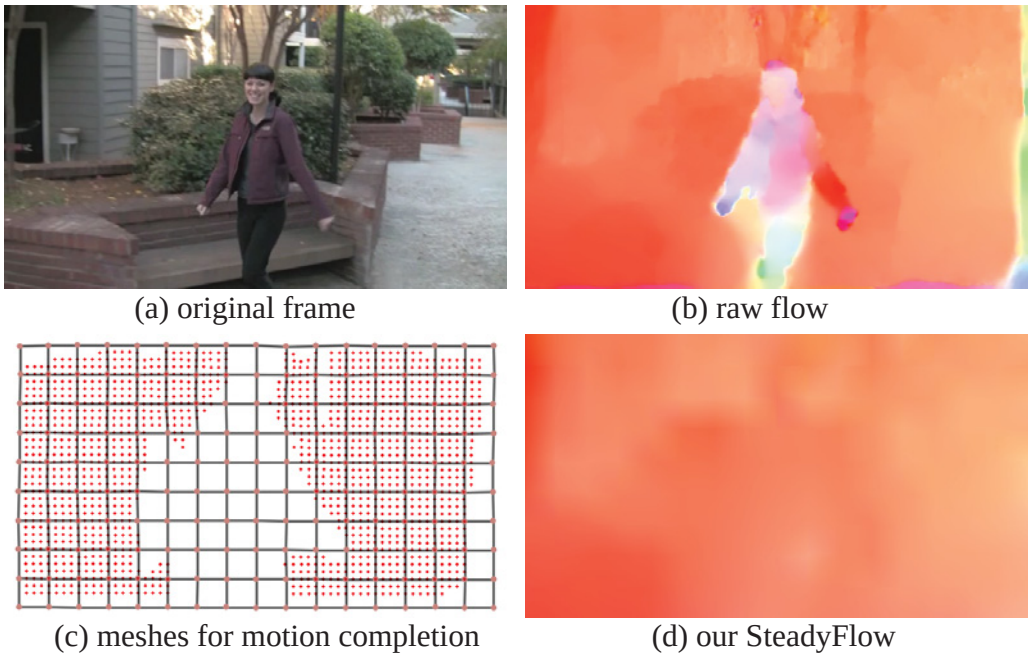
where $\mathcal{G}$ is a Gaussian filter (with default standard deviation 3) and ε uses adaptive threshold (described in Section 5.3.4).

5.3.3 Motion Completion

We collect all the pixels with discontinuous motions to form a outlier mask. Motion vectors within the mask are discarded. We then complete it in a similar way as [60] by the 'as-similar-as-possible' warping [56, 57]. Basically, we take the pixels on the mask boundary as control points, and fill in the motion field by warping 2D meshes grids with the grid size 40×40 pixels. Mathematically, it amounts to minimizing the energy $E(V) = E_d(V) + E_s(V)$. We take the same smoothness E_s as described in [56] to maintain the rigidity of the grid. The data term E_d is defined as:

$$E_d(V) = \sum\nolimits_p \mathbf{M}(p) \cdot ||V\pi_p - (p + u_p)||. \tag{5.2}$$

Here,the grids vertices are indicated by V. The vector u_p is the initial optical flow at the pixel p, such that $(p, p + u_p)$ form a pair of control points. The parameter π_p is the bilinear coordinate, e.g. $p = V_p\pi_p$, where V_p is the 4 grid vertices enclosing p. For more detailed explanation and justification, please refer to [60, 56]. This energy is minimized by solving a sparse linear equations system. We use bilinear interpolation to compute the motion vector of every pixel according to the motion of the grid vertices.



(a) original frame (b) raw flow

(c) meshes for motion completion (d) our SteadyFlow

Figure 5.7: Example of motion completion. (a) A frame from input video. (b) The raw optical flow. (c) Warped mesh estimated from background samples. The white region shows the outlier mask. (d) SteadyFlow after rewrite the discontinuous motion vectors.

Figure 5.7 shows the estimated SteadyFlow. The missing regions in the flow field (white regions in Figure 5.7 (c)) corresponds to dynamic objects,depth edges (e.g. flows on tree branches) and image boundary pixels with inaccurate raw optical flows. The motion vectors in the missing regions are interpolated from their neighboring pixels. In this way, we generate the SteadyFlow as shown in Figure 5.7 (d).

The raw optical flow field might also be smoothed by strong Gaussian smooth. However, Gaussian smooth propagates the foreground motion to background pixels. This makes the frame registration fail at background and causes strong temporal wobble artifacts in the stabilized video.

5.3.4 Iterative Refinement

Note that our temporal analysis for the estimation of outliers mask requires a stable video. As shown in Figure 5.19 (c) and (d), the trajectories generated on the original shaky video is discontinuous everywhere. In practice, we obtain an initial outlier mask $\mathbf{M}_t$ estimated from the shaky video only by spatial analysis of discontinuous flow vectors. Then we apply an iterative scheme to alternatively refine the outlier mask $\mathbf{M}_t$. At each iteration, the first step is to exclude outliers and fill in the missing regions of the input SteadyFlow according to the mask $\mathbf{M}_t$. The motion completion is described in Section 5.3.3. The second step is to stabilize the SteadyFlow, which will be described in Section 5.4. In the third step, the stabilized SteadyFlow is then used to further refine $\mathbf{M}_t$ by temporal analysis of discontinuous flow vectors as described in Section 5.3.2. Since our temporal analysis is more suitable for stable videos, we may consider adaptive threshold $(1 + \alpha^{1/n})\varepsilon$ used in Equation 5.1 to assign a conservative threshold in the beginning. Here, n is the iteration index and $\alpha = 20, \varepsilon = 0.2$ is used in our experiment. We iterate the whole three steps to finally generate the stabilized result. We use 5 iterations in our experiments empirically.

5.4 Pixel Profiles based Stabilization

We here derive the stabilization algorithm that smoothes the pixel profiles extracted from the SteadyFlow. Let $\mathbf{U}_t$, $\mathbf{S}_t$ be the SteadyFlow estimated from frame t to frame $t-1$ in the input video and stabilized video respectively. The smoothing is achieved

by minimizing the following objective function similar to [61]:

$$\mathcal{O}(\{\mathbf{P}_t\}) = \sum_t \left(\|\mathbf{P}_t - \mathbf{C}_t\|^2 + \lambda \sum_{r \in \Omega_t} w_{t,r} \|\mathbf{P}_t - \mathbf{P}_r\|^2 \right), \tag{5.3}$$

where $\mathbf{C}_t = \sum_t \mathbf{U}_t$ is the field of accumulated motion vectors of the input video. Similarly, we have $\mathbf{P}_t = \sum_t \mathbf{S}_t$ of the stabilized video. The first term requires the stabilized video staying close to its original to avoid excessive cropping, while the second term enforces temporal smoothness.

There are three differences from path optimization in [61]. First, since SteadyFlow itself enforces strong spatial smoothness, we do not require any spatial smoothness constraint in Eqn 5.3. Second, the weight $w_{t,r}$ only involves the spatial Gaussian function $w_{t,r} = exp(-||r - t||^2/(\Omega_t/3)^2)$ rather than a bilateral weight. To adaptively handle different motion magnitudes, we adopt an adaptive temporal window Ω_t in our smoothing (to be discussed in Section 5.4.1). Third, the $\mathbf{P}$ and $\mathbf{C}$ here are non-parametric accumulated motion vectors instead of parametric models (e.g. homographies).

Likewise, we can further obtain iterative solution by:

$$\mathbf{P}_t^{(\xi+1)} = \frac{1}{\gamma} \left(\mathbf{C}_t + \lambda \sum_{r \in \Omega_t, r \neq t} w_{t,r} \mathbf{P}_r^{(\xi)} \right), \tag{5.4}$$

where the scalar $\gamma = 1 + \lambda \sum_r w_{t,r}$ and ξ is an iteration index (by default, $\xi = 10$). After optimization, we will warp the original input video frame to the stabilized frame by a dense flow field $\mathbf{B}_t = \mathbf{P}_t - \mathbf{C}_t$. We can further derive the relationship between

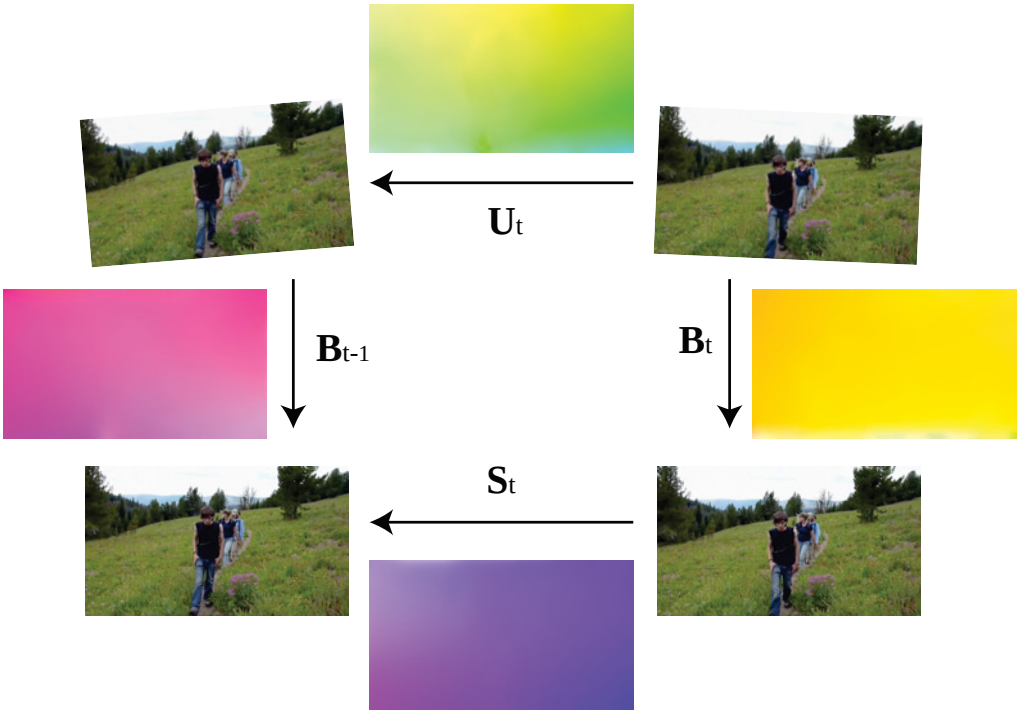


Figure 5.8: Flow fields in our stabilization. $\mathbf{U}_t$ present SteadyFlow between input frames; $\mathbf{B}_t$ represent warping from input frame to output frame; $\mathbf{S}_t$ represent flow field between output frames.

$\mathbf{B}_t$ and $\mathbf{U}_t$, $\mathbf{S}_t$ as:

$$\mathbf{U}_t + \mathbf{B}_{t-1} = \mathbf{B}_t + \mathbf{S}_t \Rightarrow \mathbf{S}_t = \mathbf{U}_t + \mathbf{B}_{t-1} - \mathbf{B}_t. \tag{5.5}$$

5.4.1 Adaptive Window Selection

Our smoothing technique requires a feature trajectories to be similar to its corresponding pixel profile within the temple window Ω_t. We adaptively adjust the size of Ω_t to deal with motion velocity changes in the video. Specifically, as shown in Figure 5.9, the SteadyFlow is assumed to be spatially smooth within a window (denoted by the

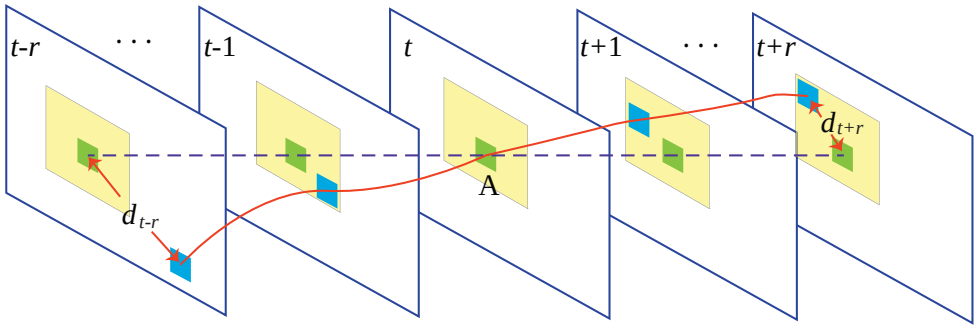


Figure 5.9: Estimation of adaptive temporal window size Ω_t in Equation 5.3. The window size Ω_t is selected such that the feature trajectory (denoted by the red line) is always within the predetermined yellow box.

yellow box) of the size$(2\tau + 1) \times (2\tau + 1)$, centered at pixel A. Within the window, smoothing the feature trajectory (denoted by the solid red line) can be well approximated by smoothing the pixel profile (denoted by the dish blue line). Once the trajectory goes outside the window, e.g. $d_{t-r} > \tau$ ($\tau = 20$ in our implementation), it would introduce non-negligible errors to the approximation. So we estimate Ω_t for each pixel in a pixel profile to ensure the feature trajectory started at that pixel is within $(2\tau + 1) \times (2\tau + 1)$ for all frames in Ω_t. The feature trajectory here is approximated by tracing the optical flows. For instance, in Figure 5.9, the window for point A is $\Omega_t(A) = [t - 1, t + r]$. To avoid spatial distortion, it is necessary to choose a global smooth window Ω_t for all pixels in the frame t. So we take the intersection of the windows at all pixels to determine the final temporal support for frame t. With the help of dynamic window, we can handle videos with quick camera motion e.g. quick rotation, fast zooming.

5.5 Results

We evaluated our method on some challenging examples from publicly available videos in prior publications to facilitate comparisons. These example videos include large parallax, dynamic objects, large depth variations, and rolling shutter effects.

Our system takes 1.5 second to process a video frame (640×360 pixels) on a laptop with 2.3GHz CPU and 4G RAM. The computation bottleneck is the optical flow estimation (1.1 second per frame), which could be significantly speed up by GPU implementations. Our outliers mask estimation takes 0.34 second on each frame. It is independent per-pixel computation and can be parallelized easily.

Videos with Large Dynamic Objects This is a challenging case for previous 2D video stabilization methods. A large portion of corresponding image features are on the foreground moving objects. Previous methods often rely on RANSAC to exclude these points to estimate the background 2D motion.

Figure 5.10 shows a synthetic example. We compared our method with a simple 2D technique that adopts homography fitting with RANSAC for motion estimation. In Figure 5.10 (a), we can see that RANSAC cannot exclude all the outliers, which cause distortions in the results as shown in (c). In comparison, our SteadyFlow estimation can exclude all the undesirable motion vectors on the foreground object (see Figure 5.10 (b)) and produce better stabilization result in (d). To further know how our SteadyFlow estimation extract outlier masks for this example, Figure 5.11 shows the intermediate masks at each iteration.

In addition, we borrow four videos (shown in Figure 5.12) with remarkable dynamic objects from [57], [35] and [61], which are reported as failure cases. The large moving object (a person) in the first video (shown in Figure 5.12 (a)) breaks feature

trajectories, and makes feature-track-based method (like [57]) fail. The examples in Figure 5.12 (b) and (c) clearly consist of two motion layers. For both examples, our method can identify distracting foreground objects despite their large image size. This ensures a successful SteadyFlow estimation and superior stabilization results. The recent 'Bundled-Paths' method [61] fits a 2D grid of homographies to model more complicated 2D motion. This method enforces stronger spatial smoothness at dynamic scenes, which reduces their model representation capability. Thus, it produces artifacts on the example shown in Figure 5.12 (d). In comparison, our SteadyFlow is powerful to exclude dynamic objects and can maintain the ability of modelling complicated motion. As a result, we can produce better results.

Videos with Large Depth Change We further evaluate our method on two videos with large depth changes, one video come from [56] and another captured by ourselves. Our 2D method achieved results of similar visual quality to 3D method. The video thumbnails are shown in Figure 5.13. We compared our results with that of a traditional 2D method [64] (using our implementation). As can be seen from the accompany video, the results from [64] contain jitters at some image regions. We further compare with indoor videos captured by Kinect[60].

Videos Captured by Kinect An additional depth camera simplifies the stabilization problem as demonstrated in [60]. It produces superior stabilization results to other methods on challenging indoor videos with large depth variations. We applied our method on two videos from that paper (see sample frames in Figure 5.14). For both examples, we only used the RGB video as the input and achieved comparable results as reported in [60].

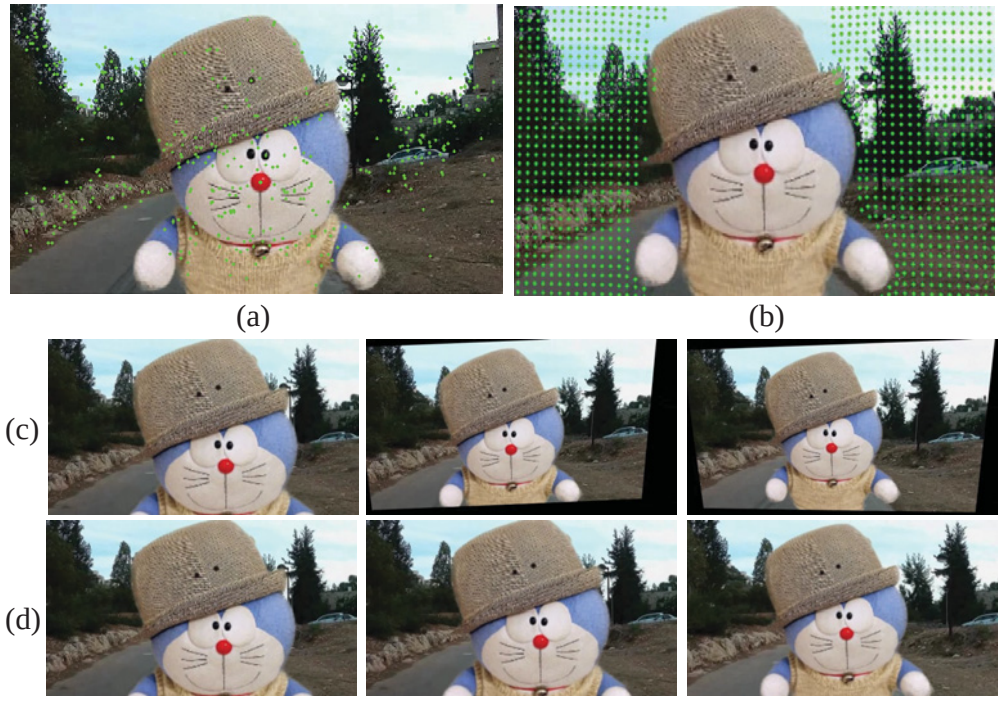


Figure 5.10: Comparison with single homography based stabilization. (a) Inliers after RANSAC based homography fitting. (b) Inlier motion vectors after our outlier mask detection. (c) and (d) are results from the single homography based method and our method respectively.

Videos with Rolling Shutter Effects Rolling shutter effects of CMOS sensors cause spatial variant motions in videos. Our method can model rolling shutter effects as spatially variant high frequency jitters. It can simultaneously rectify rolling shutter effects when smoothing camera shakes. Figure 5.15 shows two rolling shutter videos borrowed from [38]. Our method produced similar quality as other state-of-art techniques [5, 49, 38, 61].

Comparison with State-of-art System We further compared our system with two well-known commercial systems on our captured videos. One system is the YouTube Stabilizer, which is built upon the L_1-optimization method [39] and the homography

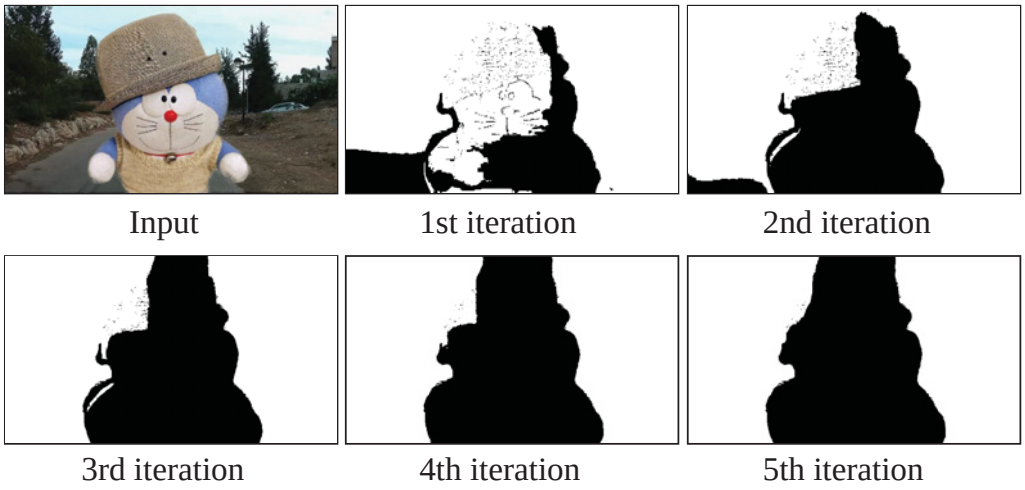


Figure 5.11: Estimated masks during each iteration of optimization on a synthetic example.

mixture method [38]. We uploaded our videos to YouTube and downloaded the automatically stabilized results. Another system is the Adobe After Effects CS6 'Warp Stabilizer', which is based on the subspace stabilization method [57]. Since it is an interactive tool, we try our best to generate results with the best perceptual quality.

Figure 5.16 shows the comparison with YouTube Stabilizer. We can see remarkable structure distortions at the pole, which has discontinuous depth changes. In comparison, our SteadyFlow estimation masks out these depth changes and fill in by the neighboring motions. Thus our result is stable and free from distortions.

Figure 5.17 shows the comparison with the 'Warp Stabilizer' in After Effects CS6. In this example, the moving train makes the feature-trajectory-based subspace analysis fail. As a result, shearing/skewing distortions are visible in their result. Our SteadyFlow estimation excludes motion vectors on the train to obtain a spatially coherent motion field for stabilization. Our result is free from distortions, though it might not be physically correct.

Figure 5.12: Failure examples reported in (a) and (b) Subspace stabilization [57], (b) Epipolar [35], (d) Bundled-Paths [61].

Limitation During the experiment, we noticed that the size of the foreground is crucial to a successful result. Our spatial-temporal analysis fails to distinguish foreground and background when videos contain dominant foreground objects. These objects consistently occupy more than half area of a frame and exsit for a long time. The stabilization will be applied on the foreground instead of background, or keep switching. Figure 5.18 shows two failure cases.

Figure 5.13: Two videos with large depth change for the comparison with traditional 2D stabilization.

Figure 5.14: Two test videos borrowed from [60].

Figure 5.15: Two rolling shutter videos borrowed from [38].

(a) YouTube result (b) our result

Figure 5.16: Comparison with YouTube Stabilizer. The red arrow indicates structure distortions in YouTube results.

(a) 'Warp Stabilizer' result (b) our result

Figure 5.17: Comparison with Adobe After Effects CS6 'Warp Stabilizer'. We can notice the global shearing/skewing in 'Warp Stabilizer' results.

Figure 5.18: Failure cases. Videos contain dominant foreground.

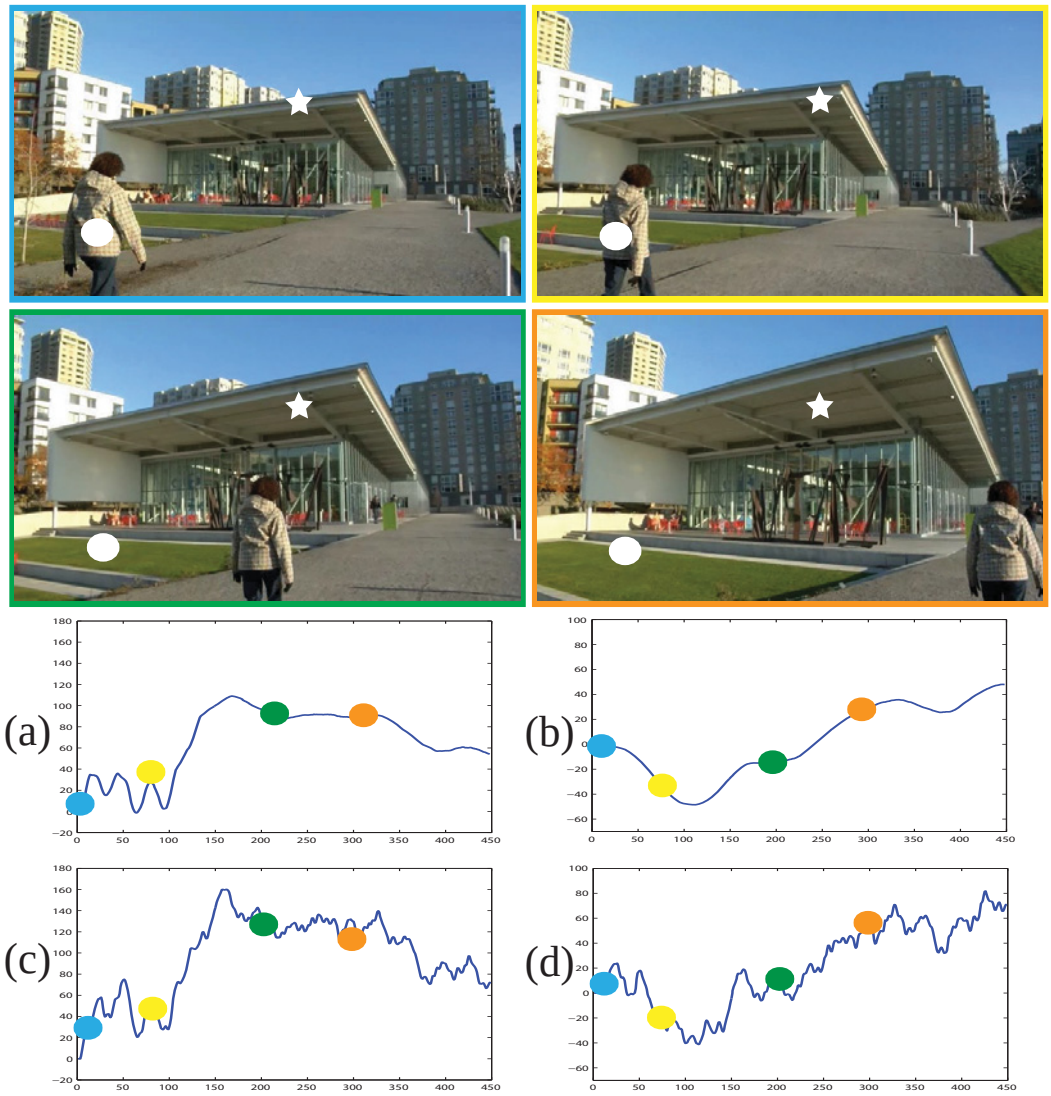


Figure 5.19: We identify discontinuous motion vectors by analyzing if the trajectory of accumulated motion vectors on a pixel profile is temporally smooth. We show four frames from a stabilized video. (a) and (b) are the trajectories of the accumulated motion vectors evaluated at the pixels marked by white dot and white star. (c) and (d) are the trajectories at the corresponding positions on the input video. The temporal locations of these 4 frames are denoted in the trajectories by dots with the same color as the frame border.

5.6 Conclusion

We propose a novel motion representation, SteadyFlow, for video stabilization. Due to the strong spatial coherence in the SteadyFlow, we can simply smooth each motion profile independently without considering the spatial smoothness [61] or subspace constraint [60]. Our method is more robust than previous 2D or 3D methods. Its general motion model allows stabilizing challenging videos with large parallax, dynamic objects, rolling-shutter effects, etc.

Chapter 6

Conclusions

6.1 Chapter Summaries

In this thesis, we have proposed several solutions to the problem of video stabilization. In chapter 1, we introduced the problem of video stabilization and discussed the challenges related to this topic. We showed that the large depth variation and large moving objects were challenging issues for camera motion estimation. We demonstrated the importance of handling quick camera motions. We also introduced the rolling shutter effects and motion blur as two common accompany issues related to video stabilization. We further demonstrated the kind of artifacts caused by these challenges on various video stabilization methods.

According to the adopted motion models, video stabilization methods can be categorized into 2D, 3D and 2.5D . In chapter 2, we revisited most related video stabilization approaches based on these categories. The contribution of this thesis consists of three novel methods, video captured by a depth camera, bundled camera paths and SteadyFlow for video stabilization, with the first targets on depth camera and the latter

two focus on casual shot videos by traditional hand-held devices. They are presented in chapter 3,4 and 5.

Chapter 3 presented the method for video stabilization with a depth camera. We studied two challenges in video stabilization, namely large depth changes in the indoor environment which make 2D motion model imprecise and tracking failures which cause 3D stabilization to fail. We proposed to use an additional depth sensor such as a Kinect camera to address these challenging cases. Though the depth image is noisy, incomplete and low resolution, it facilitates both camera motion estimation and frame warping, which makes the video stabilization a better posed problem. We combined color and depth images to robustly compute 3D camera motion. We matched 2D features between neighboring frames, and used their depths to estimate relative camera motion. We then smoothed the recovered 3D camera trajectories following cinematography principles. For the novel view synthesize, we generated a dense non-linear motion field to combine 3D projection and 2D image warping.

Chapter 4 presented a new 2D video stabilization method with a bundled camera paths model. The proposed method can simultaneously generate comparable results to 3D methods while keeping merits of 2D methods. We proposed to use 'as-similar-as-possible' warping approach to model the motion between neighboring frames. We divided frames into cells blocks, each of which contains its own camera path, thus all the cells form a bundled camera paths on the whole video. This spatial variant motion representation could handle scenes with large depth variation. We also introduced a path smoothing method to handle quick camera motion(e.g., quick camera rotation and zooming). This adaptive-based smoothing strategy could find a good balance between stability and cropping size. The evaluation on a large variety of consumer videos demonstrated the merits of our method.

Chapter 5 presented a novel motion model, SteadyFlow, to represent the motion between neighboring frames for video stabilization. A Steadyflow was a special optical flow by enforcing strong spatial coherence. We proposed the concept of a pixel profile which are motion vectors collected at the same pixel location in the SteadyFlow over time. With strong spatial smoothness, a feature trajectory could be well approximated by a pixel profile. Thus we could smooth pixel profiles to stabilize a video. Compared to feature trajectories, pixel profiles were spatially and temporally dense and could be smoothed independently without considering the spatial smoothness or subspace constraints. We estimated optical flow between neighboring frames,which is followed by a special-temporal analysis to exclude discontinuous depth and large moving objects. We then inpainted these regions by the surrounding optical flow to obtain the SteadyFlow. We demonstrated the advantages of the SteadyFlow by stabilizing challenging consumer videos with large parallax, dynamic objects and rolling-shutter effects,etc.

6.2 Future Research

There are several future research directions for the work presented in the thesis. One is robustly handling large moving objects. When video contains dominate foreground objects, existing method fails to distinguish foreground and background. These objects consistently occupy more than half area of a frame and exist for a long time. The stabilization will be applied on the foreground instead of the background. Advanced motion segmentation should be incorporated to work together with video stabilization or some user interaction[4] are also favored to this problem.

Video blurring can severely influence the quality of feature matching or tracking. It

also damages the quality of a stabilized video. Video deblurring[19] targets on turning blurry frames into sharp ones. In fact, the blurring is "amplified" when viewed in the stabilized results. This is mainly due to the change of camera motion with unchanged frame blur directions. In general, video stabilization and video deblurring are two related problems. There are potentials that these two problems can be solved together.

It is also worth to explore the possibility to stabilize a video with the help of hardware devices(e.g., gyroscopes[49]). Nowadays, video stabilization methods are either purely based on software as a post processing method or based on hardware for real-time applications. We can design some hybrid approaches to combine the benifits from both. For realtime applications, we need to design a new path smoothing strategy because we can only look at previous camera paths with unknown future frames.

Our mesh-based motion model described in the Bundled camera path[61] can be used for other applications. In general, this motion model can be applied for problems requiring image registration. For example, image/video denoising[13, 14, 16, 18, 54, 94, 93], super resolution[31, 78, 59, 55, 6, 28, 73, 8], content-aware resizing[88, 83, 84, 87, 89], HDR imaging[48, 36, 72, 80, 47] mosaics/panoramas[75, 77, 2, 3, 10, 30, 52].

On this note, we would like to conclude this thesis.

Bibliography

[1] S. Agarwal, N. Snavely, I. Simon, S. M. Seitz, and R. Szeliski. Building rome in a day. In *IEEE International Conference on Computer Vision(ICCV)*, 2009. 13

[2] A. Agarwala, M. Agrawala, M. Cohen, D. Salesin, and R. Szeliski. Photographing long scenes with multi-viewpoint panoramas. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 25(3):853–861, 2006. 97

[3] A. Agarwala, M. Dontcheva, M. Agrawala, S. Drucker, A. Colburn, B. Curless, D. Salesin, and M. Cohen. Interactive digital photomontage. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 23(3):294–302, 2004. 97

[4] J. Bai, A. Agarwala, M. Agrawala, and R. Ramamoorthi. User-assisted video stabilization. In *Computer Graphics Forum(CGF)*, volume 33, pages 61–70, 2014. 5, 96

[5] S. Baker, E. P. Bennett, S. B. Kang, and R. Szeliski. Removing rolling shutter wobble. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2010. 19, 87

[6] S. Baker and T. Kanade. Limits on super-resolution and how to break them. *IEEE Transactions on Pattern Analysis and Machine Intelligence(TPAMI)*, 24(9):1167–1183, 2002. 97

[7] H. Bay, A. Ess, T. Tuytelaars, and L. Van Gool. Speeded-up robust features (surf). *Computer Vision Image Understanding(CVIU)*, 110(3):346–359, 2008. 25, 28, 55

[8] M. Ben-Ezra, A. Zomet, and S. K. Nayar. Video super-resolution using controlled subpixel detector shifts. *IEEE Transactions on Pattern Analysis and Machine Intelligence(TPAMI)*, 27(6):977–987, 2005. 97

[9] I. N. Bronshtein and K. A. Semendyayev. *Handbook of Mathematics*. Springer-Verlag, 1997. 50, 54

[10] M. Brown and D. G. Lowe. Automatic panoramic image stitching using invariant features. *International Journal of Computer Vision(IJCV)*, 74(1):59–73, 2007. 97

[11] T. Brox, A. Bruhn, N. Papenberg, and J. Weickert. High accuracy optical flow estimation based on a theory for warping. In *European Conference on Computer Vision(ECCV)*, 2004. 46

[12] T. Brox and J. Malik. Object segmentation by long term analysis of point trajectories. In *European Conference on Computer Vision(ECCV)*, 2010. 77, 78

[13] A. Buades, B. Coll, and J.-M. Morel. A non-local algorithm for image denoising. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2005. 97

[14] A. Buades, B. Coll, and J.-M. Morel. A review of image denoising algorithms, with a new one. *Multiscale Modeling & Simulation*, 4(2):490–530, 2005. 97

[15] C. Buehler, M. Bosse, and L. McMillan. Non-metric image-based rendering for video stabilization. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2001. 13, 15, 25

[16] P. Chatterjee, N. Joshi, S. B. Kang, and Y. Matsushita. Noise suppression in low-light images through joint denoising and demosaicing. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2011. 97

[17] B.-Y. Chen, K.-Y. Lee, W.-T. Huang, and J.-S. Lin. Capturing intention-based full-frame video stabilization. *Computer Graphics Forum(CGF)*, 27(7):1805–1814, 2008. 15

[18] J. Chen and C.-K. Tang. Spatio-temporal markov random field for video denoising. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2007. 97

[19] S. Cho, J. Wang, and S. Lee. Vdeo deblurring for hand-held cameras using patch-based synthesis. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 31(4):64:1–64:9, 2012. 6, 97

[20] S. Cho, J. Wang, and S. Lee. Video deblurring for hand-held cameras using patch-based synthesis. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 31(4), 2012. 15, 66

[21] T. Crivelli, P.-H. Conze, P. Robert, M. Fradet, and P. Pérez. Multi-step flow fusion: towards accurate and dense correspondences in long video shots. In *British Machine Vision Conference(BMVC)*, 2012. 75

[22] Z. Dong, G. Zhang, J. Jia, and H. Bao. Keyframe-based real-time camera tracking. In *IEEE International Conference on Computer Vision(ICCV)*, 2009. 11

[23] R. Dragon, Hannover, B. Rosenhahn, and J. Ostermann. Multi-scale clustering of frame-to-frame correspondences for motion segmentation. In *European Conference on Computer Vision(ECCV)*, 2012. 7, 77

[24] M. A. Fischler and R. C. Bolles. Random sample consensus: a paradigm for model fitting with applications to image analysis and automated cartography. *Communications of ACM*, 24(6):381–395, 1981. 46

[25] P.-E. Forssén and E. Ringaby. Rectifying rolling shutter video from hand-held devices. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2010. 19

[26] P.-E. Forssén and E. Ringaby. Efficient video rectification and stabilization of cell-phones. *International Journal of Computer Vision(IJCV)*, 96:335–352, 2012. 19

[27] J.-M. Frahm, P. Georgel, D. Gallup, T. Johnson, R. Raguram, C. Wu, Y.-H. Jen, E. Dunn, B. Clipp, S. Lazebnik, and M. Pollefeys. Building rome on a cloudless day. In *European*

Conference on Computer Vision(ECCV), 2010. 13

[28] R. Fransens, C. Strecha, and L. Van Gool. Optical flow based super-resolution: A probabilistic approach. *Computer Vision Image Understanding(CVIU)*, 106(1):106–115, 2007. 97

[29] Y. Furukawa, B. Curless, S. M. Seitz, and R. Szeliski. Towards internet-scale multi-view stereo. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2013. 13

[30] J. Gao, S. J. Kim, and M. S. Brown. Constructing image panoramas using dual-homography warping. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2011. 97

[31] D. Glasner, S. Bagon, and M. Irani. Super-resolution from a single image. In *IEEE International Conference on Computer Vision(ICCV)*, 2009. 97

[32] M. L. Gleicher and F. Liu. Re-cinematography: Improving the camera dynamics of casual video. In *ACM Multimedia*, 2007. 12, 15, 23, 66

[33] M. L. Gleicher and F. Liu. Re-cinematography: Improving the camerawork of casual video. *ACM Transactions on Multimedia Computing, Communications, and Applications*, 5(1), 2008. 12, 15

[34] A. Goh and R. Vidal. Segmenting motions of different types by unsupervised manifold clustering. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2007. 18

[35] A. Goldstein and R. Fattal. Video stabilization using epipolar geometry. *ACM Transactions on Graphics(TOG)*, pages 1–10, 2012. xi, xii, 1, 2, 12, 17, 60, 65, 69, 71, 85, 89

[36] M. Granados, K. I. Kim, J. Tompkin, and C. Theobalt. Automatic noise modeling for ghost-free hdr reconstruction. *ACM Transactions on Graphics(TOG) (Proceedings of*

SIGGRAPH), 32(6):201, 2013. 97

[37] M. Grundmann. Computational video: Post-processing methods for stabilization,retargeting and segmentation. *Doctoral Thesis. Georgia Institute of Technology*, 2013. 3, 16

[38] M. Grundmann, V. Kwatra, D. Castro, and I. Essa. Calibration-free rolling shutter removal. In *IEEE International Conference on Computational Photography(ICCP)*, 2012. x, xi, xii, 3, 19, 20, 55, 56, 57, 58, 61, 69, 87, 88, 90

[39] M. Grundmann, V. Kwatra, and I. Essa. Auto-directed video stabilization with robust l1 optimal camera paths. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2011. x, 1, 15, 16, 24, 27, 29, 30, 34, 36, 38, 60, 61, 87

[40] G. Hanning, N. Forslów, P.-E. Forssén, E. Ringaby, D. Tórnqvist, and J. Callmer. Stabilizing cell phone video using inertial measurement sensors. 19

[41] R. Hartley and A. Zisserman. *Multiple View Geometry in Computer Vision*. Cambridge University Press, New York, NY, USA, 2 edition, 2003. 1, 12, 23, 53

[42] T. Igarashi, T. Moscovich, and J. F. Hughes. As-rigid-as-possible shape manipulation. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 24(3):1134–1141, 2005. 13, 40, 42, 43

[43] M. Irani. Multi-frame correspondence estimation using subspace constraints. *International Journal of Computer Vision(IJCV)*, 48:173–194, 2002. 17

[44] M. Irani. Multi-frame correspondence estimation using subspace constraints. *International Journal of Computer Vision(IJCV)*, 48(3):173–194, 2002. 18

[45] N. Jiang, Z. Cui, and P. Tan. A global linear method for camera pose registration. In *IEEE International Conference on Computer Vision(ICCV)*, 2013. 13

[46] N. Jiang, P. Tan, and L.-F. Cheong. Seeing double without confusion: Structure-from-motion in highly ambiguous scenes. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2012. 13

[47] N. K. Kalantari, E. Shechtman, C. Barnes, S. Darabi, D. B. Goldman, and P. Sen. Patch-based high dynamic range video. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH asia)*, 32(6):202, 2013. 97

[48] S. B. Kang, M. Uyttendaele, S. Winder, and R. Szeliski. High dynamic range video. In *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, volume 22, pages 319–325, 2003. 97

[49] A. Karpenko, D. E. Jacobs, J. Baek, and M. Levoy. Digital video stabilization and rolling shutter correction using gyroscopes. In *Stanford Computer Science Tech Report CSTR 2011-03*, 2011. 15, 19, 87, 97

[50] K.-Y. Lee, Y.-Y. Chuang, B.-Y. Chen, and M. Ouhyoung. Video stabilization using robust feature trajectories. In *IEEE International Conference on Computer Vision(ICCV)*, 2009. 17, 22, 24, 25

[51] C.-K. Liang, L.-W. Chang, and H. H. Chen. Analysis and compensation of rolling shutter effect. In *IEEE Transactions on Image Processing(TIP)*, 2008. 19

[52] W.-Y. Lin, S. Liu, Y. Matsushita, T.-T. Ng, and L.-F. Cheong. Smoothly varying affine stitching. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2011. 97

[53] C. Liu. Beyond pixels: Exploring new representations and applications for motion analysis. *Doctoral Thesis. Massachusetts Institute of Technology*, 2009. 77

[54] C. Liu and W. T. Freeman. A high-quality video denoising algorithm based on reliable motion estimation. In *European Conference on Computer Vision(ECCV)*, pages 706–719, 2010. 97

[55] C. Liu and D. Sun. On bayesian adaptive video super resolution. *IEEE Transactions on Pattern Analysis and Machine Intelligence(TPAMI)*, 2013. 97

[56] F. Liu, M. Gleicher, H. Jin, and A. Agarwala. Content-preserving warps for 3d video stabilization. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 28, 2009. 1, 13, 18, 22, 25, 31, 33, 34, 40, 42, 43, 44, 46, 60, 65, 79, 86

[57] F. Liu, M. Gleicher, J. Wang, H. Jin, and A. Agarwala. Subspace video stabilization. *ACM Transactions on Graphics(TOG)*, 30, 2011. x, xii, 1, 3, 17, 22, 24, 25, 34, 35, 38, 40, 60, 62, 65, 69, 75, 79, 85, 86, 88, 89

[58] F. Liu, Y. Niu, and H. Jin. Joint subspace stabilization for stereoscopic video. In *IEEE International Conference on Computer Vision(ICCV)*, 2013. 17

[59] S. Liu. Digital image super resolution. *Master Thesis. National University of Singapore*, 2010. 97

[60] S. Liu, Y. Wang, L. Yuan, J. Bu, P. Tan, and J. Sun. Video stabilization with a depth camera. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2012. xii, 9, 11, 79, 86, 90, 93

[61] S. Liu, L. Yuan, P. Tan, and J. Sun. Bundled camera paths for video stabilization. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 32(4), 2013. xii, 3, 9, 12, 13, 69, 82, 85, 86, 87, 89, 93, 97

[62] S. Liu, L. Yuan, P. Tan, and J. Sun. Steadyflow: spatially smooth optical flow for video stabilization. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2014. 10, 12, 13

[63] B. D. Lucas and T. Kanade. An iterative image registration technique with an application to stereo vision. In *International Joint Conference on Artificial Intelligence (IJCAI)*, pages 674–679, 1981. 25, 72, 77

[64] Y. Matsushita, E. Ofek, W. Ge, X. Tang, and H.-Y. Shum. Full-frame video stabilization with motion inpainting. *IEEE Transactions on Pattern Analysis and Machine Intelligence(TPAMI)*, 28:1150–1163, 2006. 1, 15, 22, 24, 50, 86

[65] Y. Matsushita, E. Ofek, X. Tang, and H.-Y. Shum. Full-frame video stabilization. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2005. 3, 12, 69

[66] C. Morimoto and R. Chellappa. Evaluation of image stabilization algorithms. In *IEEE International Conference on Acoustics, Speech and Signal Processing*, pages 2789 – 2792, 1998. 15

[67] J. Nakamura. *Image Sensors and Signal Processing for Digital Still Cameras*. CRC Press, Inc., 2005. 6, 12

[68] M. Narayana, A. Hanson, and E. Learned-Miller. Coherent motion segmentation in moving camera videos using optical flow orientations. In *IEEE International Conference on Computer Vision(ICCV)*, 2013. 77

[69] T. Nir, A. M. Bruckstein, and R. Kimmel. Over-parameterized variational optical flow. *International Journal of Computer Vision(IJCV)*, 76(2):205–216, 2008. 42

[70] D. Nistér. An efficient solution to the five-point relative pose problem. *IEEE Transactions on Pattern Analysis and Machine Intelligence(TPAMI)*, 26:756–777, 2004. 13

[71] S. Schaefer, T. McPhail, and J. Warren. Image deformation using moving least squares. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 25(3):533–540, 2006. 40, 42

[72] P. Sen, N. K. Kalantari, M. Yaesoubi, S. Darabi, D. B. Goldman, and E. Shechtman. Robust patch-based hdr reconstruction of dynamic scenes. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 31(6):203, 2012. 97

[73] Q. Shan, Z. Li, J. Jia, and C.-K. Tang. Fast image/video upsampling. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 27(5):153, 2008. 97

[74] J. Shi and J. Malik. Motion segmentation and tracking using normalized cuts. In *IEEE International Conference on Computer Vision(ICCV)*, 1998. 7, 77

[75] H.-Y. Shum and R. Szeliski. Construction and refinement of panoramic mosaics with global and local alignment. In *IEEE International Conference on Computer Vision(ICCV)*, 1998. 97

[76] B. M. Smith, L. Zhang, H. Jin, and A. Agarwala. Light field video stabilization. In *IEEE International Conference on Computer Vision(ICCV)*, 2009. 11, 13

[77] R. Szeliski and H.-Y. Shum. Creating full view panoramic image mosaics and environment maps. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, pages 251–258, 1997. 97

[78] Y.-W. Tai, S. Liu, M. S. Brown, and S. Lin. Super resolution using edge prior and single image detail synthesis. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, pages 2400–2407, 2010. 97

[79] W. Tan, H. Liu, Z. Dong, G. Zhang, and H. Bao. Robust monocular slam in dynamic environments. In *International Symposium on Mixed and Augmented Reality (ISMAR)*, 2013. 11

[80] M. D. Tocci, C. Kiser, N. Tocci, and P. Sen. A versatile hdr video production system. In *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, volume 30, page 41, 2011. 97

[81] C. Tomasi and T. Kanade. Shape and motion from image streams under orthography: a factorization method. *International Journal of Computer Vision(IJCV)*, 9(2):137–154, 1992. 18

[82] C. Tomasi and R. Manduchi. Bilateral filtering for gray and color images. In *IEEE International Conference on Computer Vision(ICCV)*, pages 839–846, 1998. 41, 50

[83] Y.-S. Wang, H. Fu, O. Sorkine, T.-Y. Lee, and H.-P. Seidel. Motion-aware temporal coherence for video resizing. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH asia)*, 28(5), 2009. 97

[84] Y.-S. Wang, H.-C. Lin, O. Sorkine, and T.-Y. Lee. Motion-based video retargeting with optimized crop-and-warp. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 29(4), 2010. 97

[85] Y.-S. Wang, F. Liu, P.-S. Hsu, and T.-Y. Lee. Spatially and temporally optimized video stabilization. *IEEE Trans. on Visualization and Computer Graphics(TVCG)*, 17:1354–1361, 2013. 17

[86] C. wu. Towards linear-time incremental structure from motion. In *International Conference on 3DV*, 2013. 13

[87] H. Wu, Y.-S. Wang, K.-C. Feng, T.-T. Wong, T.-Y. Lee, and P.-A. Heng. Resizing by symmetry-summarization. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH asia)*, 29(6), 2010. 97

[88] O. S. Yu-Shuen Wang, Chiew-Lan Tai and T.-Y. Lee. Optimized scale-and-stretch for image resizing. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH asia)*, 27(5), 2008. 97

[89] O. S. Yu-Shuen Wang, Jen-Hung Hsiao and T.-Y. Lee. Scalable and coherent video resizing with per-frame optimization. *ACM Transactions on Graphics(TOG) (Proceedings of SIGGRAPH)*, 30(4), 2011. 97

[90] G. Zhang, Z. Dong, J. Jia, T.-T. Wong, and H. Bao. Efficient non-consecutive feature tracking for structure-from-motion. In *European Conference on Computer Vision(ECCV)*, 2010. 11

[91] G. Zhang, W. Hua, X. Qin, Y. Shao, and H. Bao. Video stabilization based on a 3d perspective camera model. *The Visual Computer*. 25

[92] G. Zhang, X. Qin, W. Hua, T.-T. Wong, P.-A. Heng, and H. Bao. Robust metric reconstruction from challenging video sequences. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2007. 11

[93] L. Zhang, S. Vaddadi, H. Jin, and S. K. Nayar. Multiple view image denoising. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2009. 97

[94] M. Zhang and B. K. Gunturk. Multiresolution bilateral filtering for image denoising. *IEEE Transactions on Image Processing(TIP)*, 17(12):2324–2333, 2008. 97

[95] Z. Zhang. Parameter estimation techniques: A tutorial with application to conic fitting. *Image and vision Computing*, 15(1):59–76, 1997. 29

[96] Z. Zhou, H. Jin, and Y. Ma. Plane-based content-preserving warps for video stabilization. In *IEEE Conference on Computer Vision and Pattern Recognition(CVPR)*, 2013. 13